* Regular Expression

→ Language accepted by finite automata Can be easily described with simple Expression called regular expression

$10 + 11 01$ {10, 1101}

$11 + 101 + 0101$ {11, 101, 0101}

0^* {ϵ, 0, 00, 000, ...}

$(00)^*$ {ϵ, 00, 0000, 000000 ....}

$(01)^+$ {01, 0101, 010101 ....}

any string {0, 1}

→ $(0+1)^*$ {ϵ, 0, 1, 00, 01, 10, 11 ....}

← ending with 01

$(0+1)^* 01$ $\{01, 001$

* any string $\{0,1\}$ containing exactly one 0

~~$(0+1)^* 0 (0+1)^*$~~

~~$1^* 01^* + 1^* 0$~~ $1^* 0 1^*$

170

* String for alphabet $\{0,1\}$ which contain even no of 1

$(0+11)^*$

$0^* (10^* 1 0^*)^*$

* all strings for alphabet $\{0,1\}$ which has substring 00

$(0+1)^* 00 (0+1)^*$

* String over the alphabet $\{0,1\}$ whose length is 5 which has 1 in the fourth position from the left

$(0+1)(0+1)(0+1) 1 (0+1)$

* any string alphabet over $\{0,1\}$ which has 1 in the second position from the left

$(0+1)\ 1\ (0+1)^*$

* ~~str~~ {0,1} which has 1 in 3rd position from right

$(0+1)^*\ 1\ (0+1)\ (0+1)$

* {0,1} whose length maximum 5.
exactly

$(0+1+\epsilon)\ (0+1+\epsilon)\ (0+1+\epsilon)\ (0+1+\epsilon)\ (0+1+\epsilon)$

* {a,b} in which no. b's is divisible by 3

$a^*(a^*\ b\ a^*\ b\ a^*\ b\ a^*)^*$

Identifier in R.E

$\phi + R = R = R + \phi$

Qn. definition R.E.

* ϕ is R.E representing Language containing nothing { }

- ϵ is R.E representing Language containing $\{\epsilon\}$

- For any i/p symbol a , a is also R.E , representing {a}

1. If R & S are R.E representing the Language r & s repectively then
$R+S$, RS , R^* are also R.E representing the language $r \cup s$, rs , r^*

2. $\epsilon + R = R + \epsilon$

ex
$R = 01 + 01$
$\{\epsilon\} \cup \{01, 1\}$
$\{\epsilon, 01, 1\}$

3. $\epsilon R = R = R.\epsilon$
$\{\epsilon\} \{01, 1\}$
$\{01, 1\}$

4 $\phi R = \phi = R \phi$

5) $\epsilon^* = \epsilon$

6) $\phi^* = \epsilon$

7) $\phi^+ = \epsilon$

8) $R \cdot R^* = R^+ = R^* R$

9 $\epsilon + R R^* = R^*$

(10) $(R^*)^* = R^*$

11) $R^* + R^* = R^*$

12) $R^* R^* = R^*$

13. $= R \cdot R = R \cdot R$

14) $P(QP)^* = (PQ)^* P$ $P=1$ $Q=0$

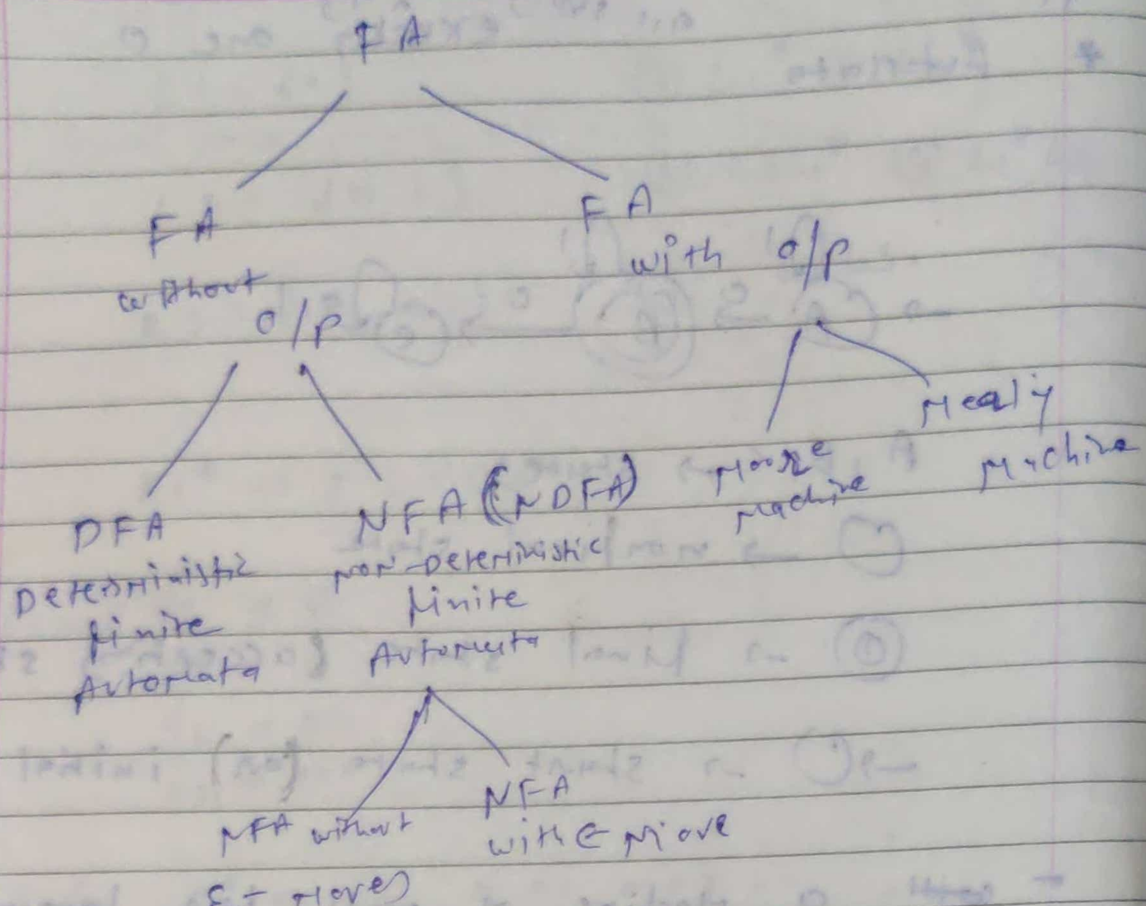
$1\,0\,1\,0\,1\,0\,1 = 1\,0\,1\,0\,1\,0\,1$

15) $(P+Q)^* = (P^* + Q^*)^*$

$= (P^* Q^*)^*$

17) $A = B + AC$ Arden's theorem

$A = BC^*$



* Deterministic FA :-
is 5 tuple
$M = (Q, \Sigma, \delta, q_0, F)$

Q. is finite set state $\rightarrow \{A, B, C\}$

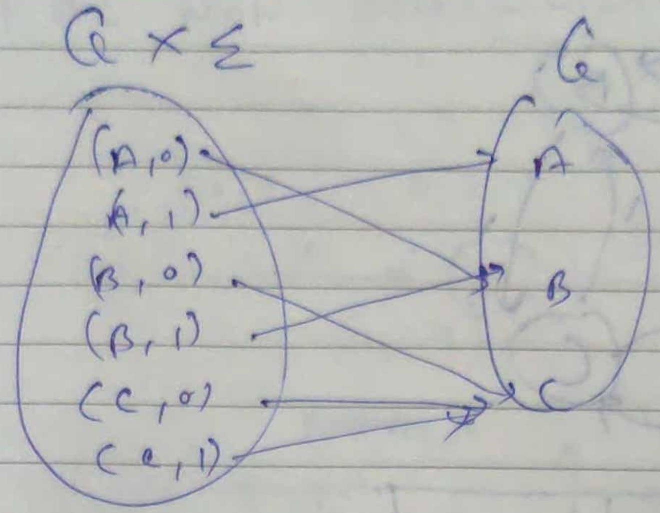
Σ is input alphabet $\rightarrow \{0, 1\}$

q_0 in Q starting state $\rightarrow \{A\}$

$F \subseteq Q$ finite set of final state $\rightarrow \{B\}$

δ is mapping from $Q \times \Sigma \rightarrow Q$

$\{ABC\} \times \{0,1\}$



$\underline{\delta}$

$\delta(A,0) = B \qquad \delta(A,1) = A$

$\delta(B,0) = C \qquad \delta(B,1) = B$

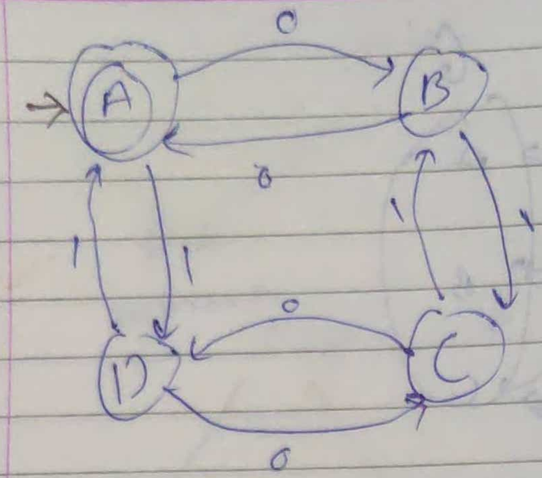
$\delta(C,0) = C \qquad \delta(C,1) = C$

* Three methods to represent a finite Autometa

1] Transition diagram
2] Mentioning of all the δ tuple individually.
3] Table representation

δ	0	1
→A	B	A
B	C	B
C	C	C

4



Method 3 to represent DFA

	0	1
→ A	B	D
B	A	C
C	D	B
D	C	A

Method 2

$Q = \{A, B, C, D\}$

$q_0 = A$

$F = \{A\}$

$\Sigma = \{0, 1\}$

$\delta =$

δ =

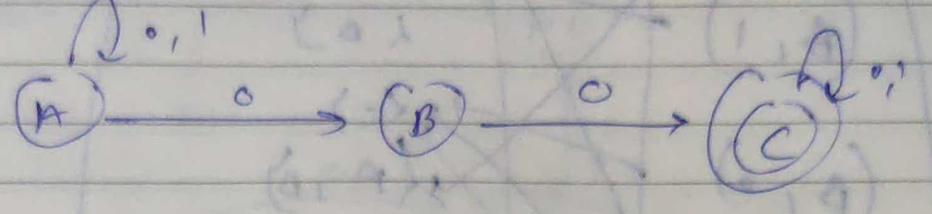
$\delta(A, 0) = B$ $\delta(A, 1) = D$

$\delta(B, 0) = A$ $\delta(B, 1) = C$

$\delta(C, 0) = D$ $\delta(C, 1) = B$

$\delta(D, 0) = C$ $\delta(D, 1) = A$

* NFA :- NON-Deterministic Finite Automata (without ϵ-moves)



DFA	NFA
→ on each state on each input symbol there is exactly one transition	on some state on some input symbol there exist more than one transmission ex $\delta(A, 0) \rightarrow \{A, B\}$
→ DFA is complete	NFA is incomplete: transmission is not define for some state on some input. in ex. $\delta(B, 1) \rightarrow \times$

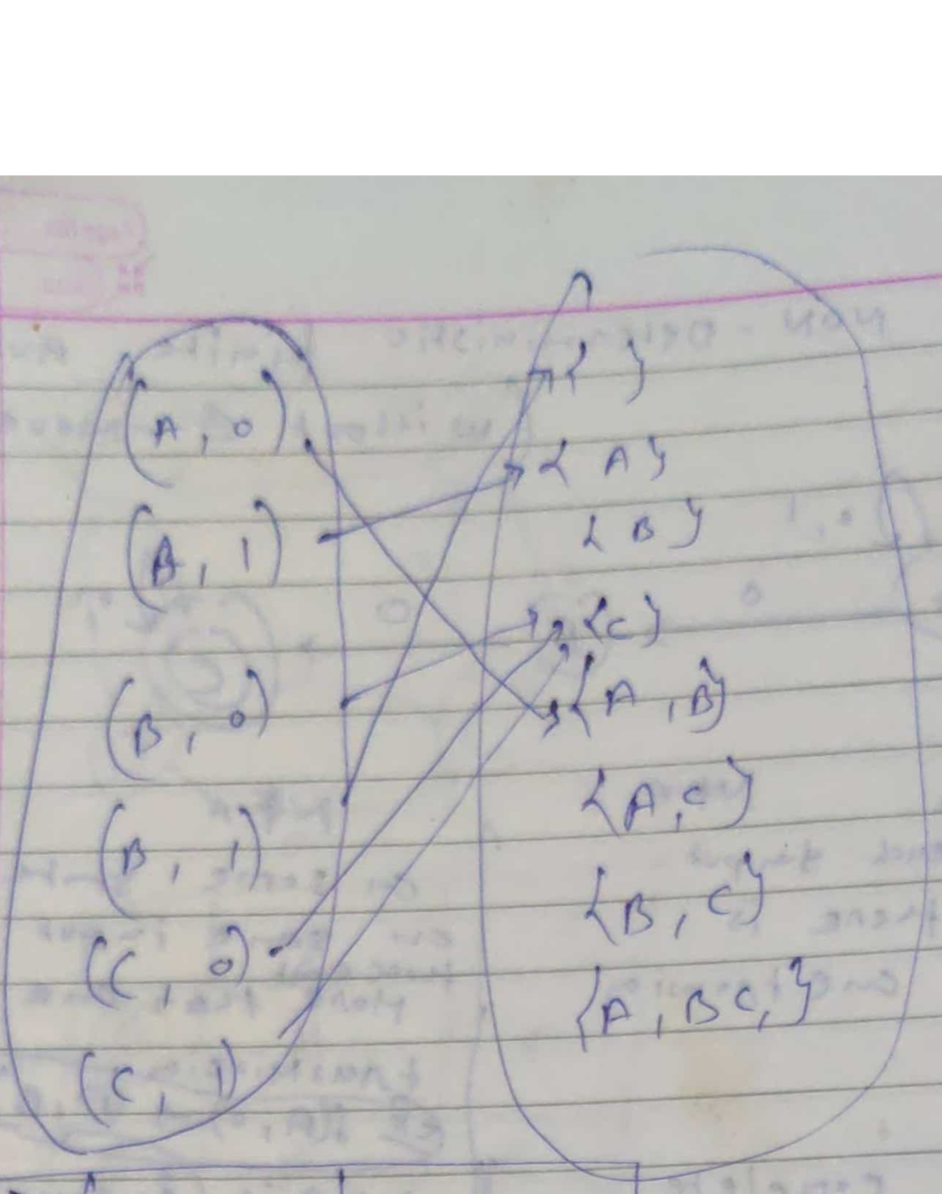
* NFA
→ 5 tuple

$M = (Q, \Sigma, \delta, q_0, F)$

where
Q, Σ, q_0, F is same is DFA

δ is mapping from $Q \times \Sigma \rightarrow 2^Q$

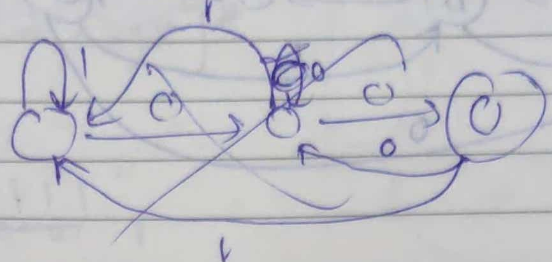
2/4

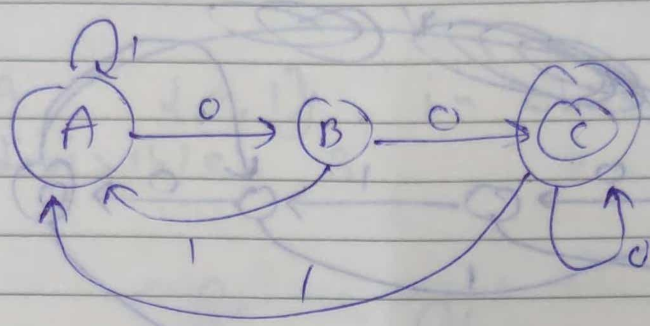


δ	0	1
→ A	{A, B}	A
B	C	φ
C	C	C

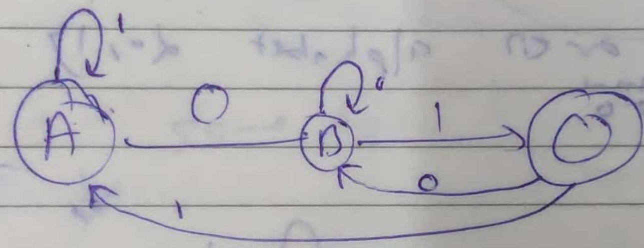
Page No.
Date: / /

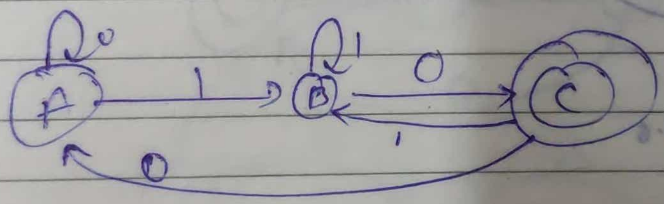
* Design DFA for language strings over alphabet {0,1} contain which are ending 00

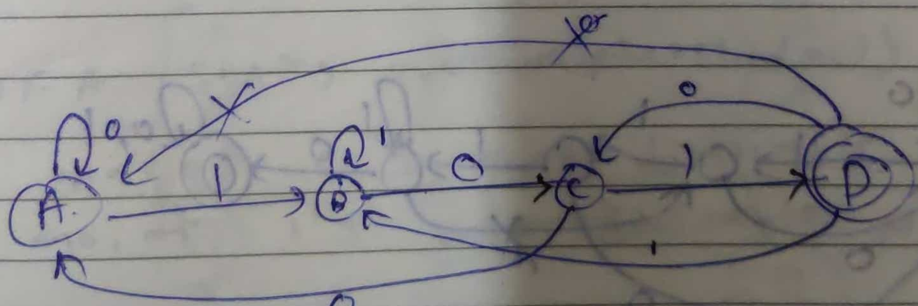




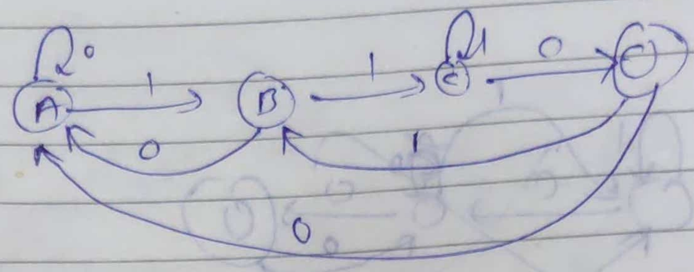
ending 01



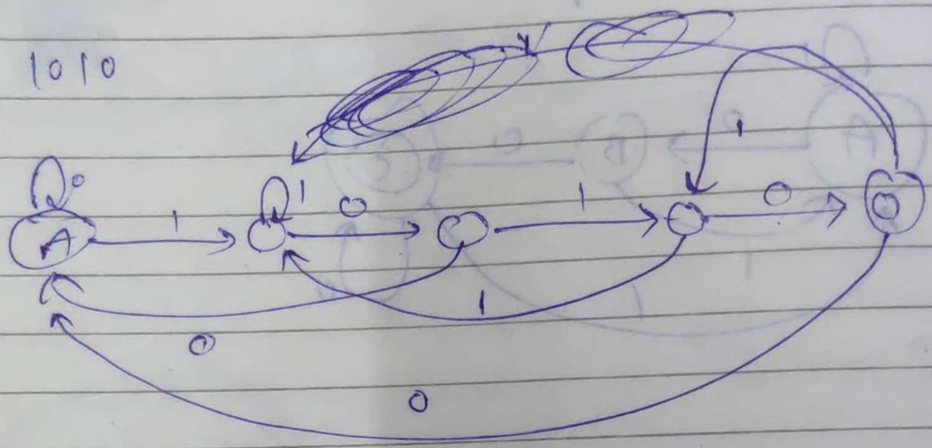




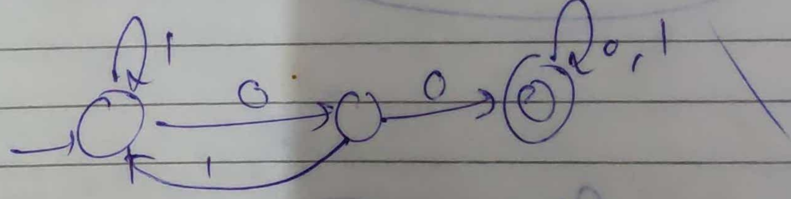
ending 110



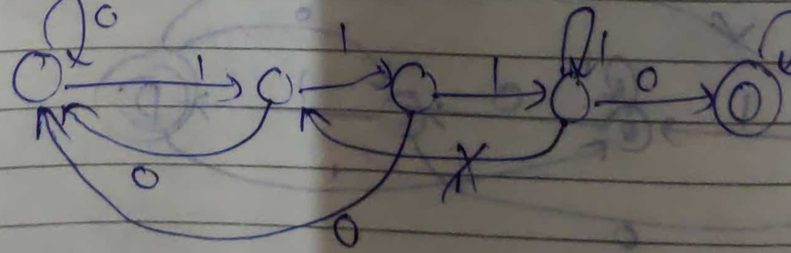
* end 1010



* DFA string over alphabet {0,1}
containing substring 00

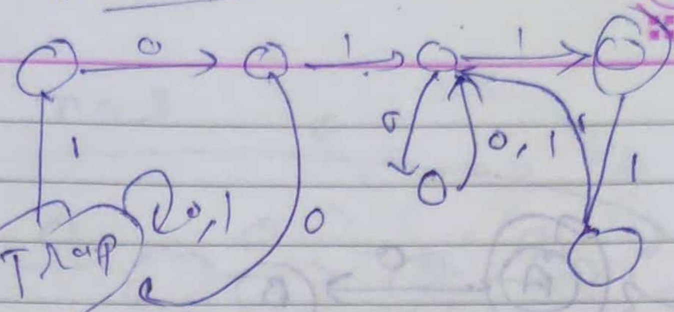


* substring 1110



11110

$0\,1\,[((01)^* + 111)^* + (00)^*]^*$

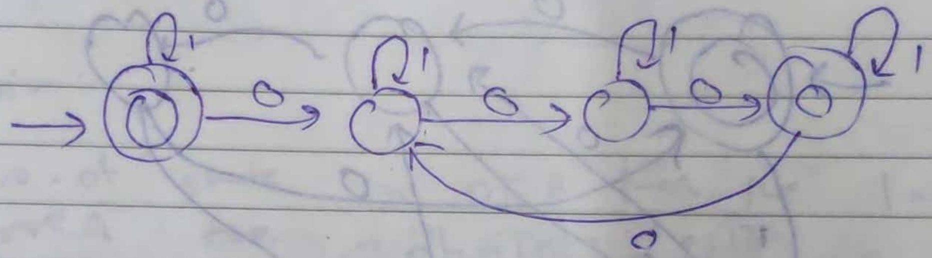


$[(((01)^* + 111)^* + (00)^*]^*$

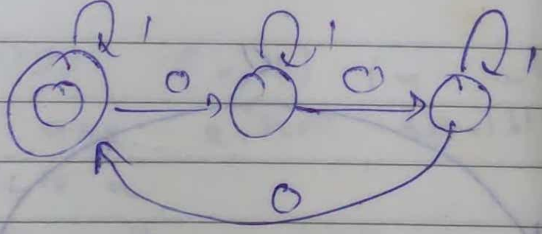
111

0 1 ~~0101~~ 1

* DFA {0, 1} in which no of 0's are divisible 3

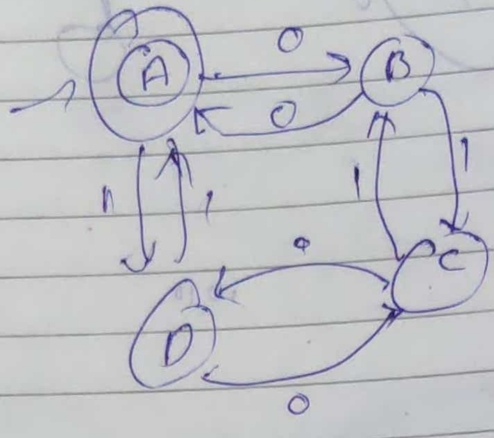


0 same as 1/,

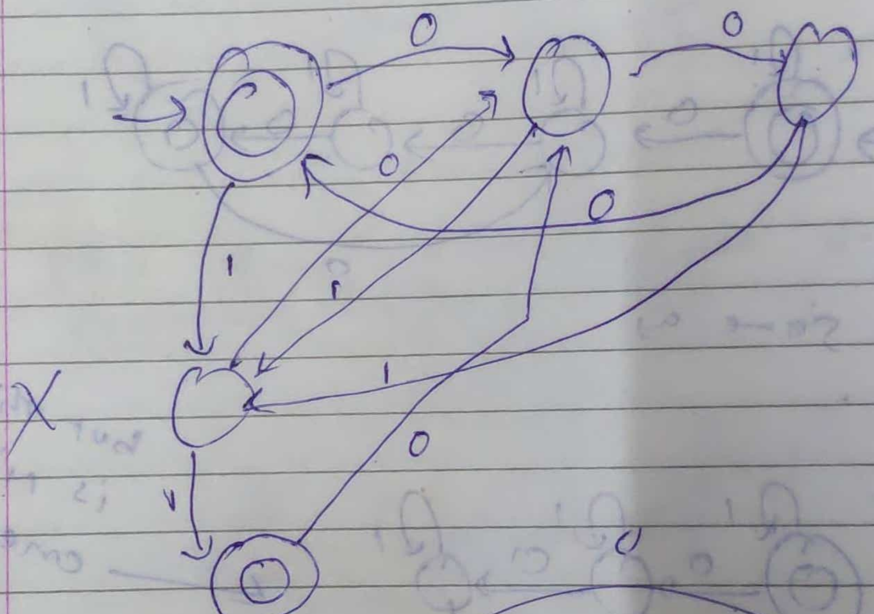


but this is minimum one

* DFA string over alphabet {0, 1} in which no. of 0's are divisible M and no. of 1's are divisable N



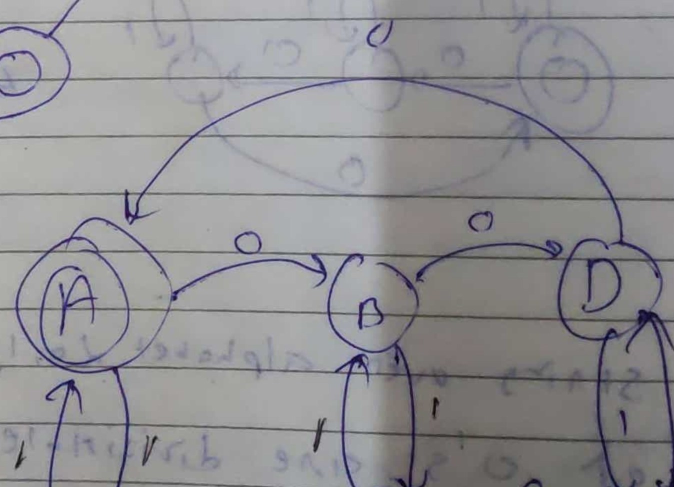
$m = 3 \qquad n = 2$



111100011

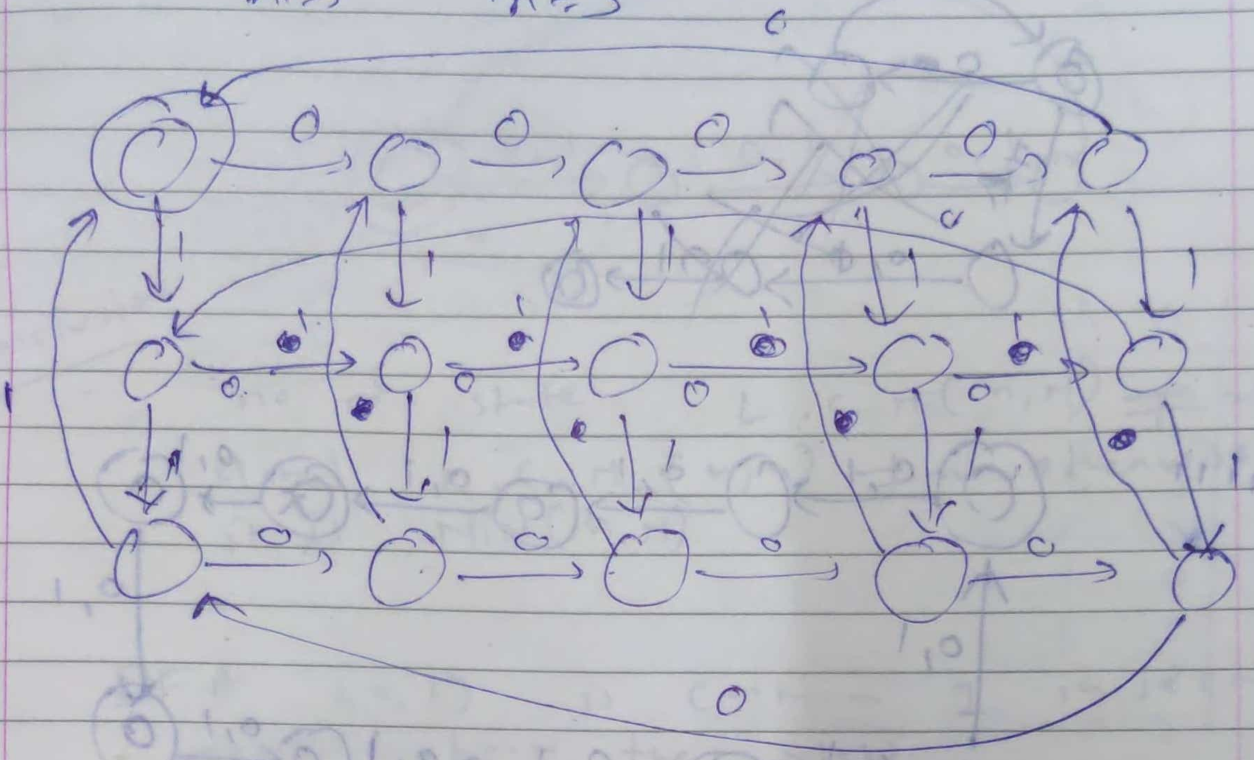
Corner

10100



10101000

m = 5 n = 3



Conclusion :-

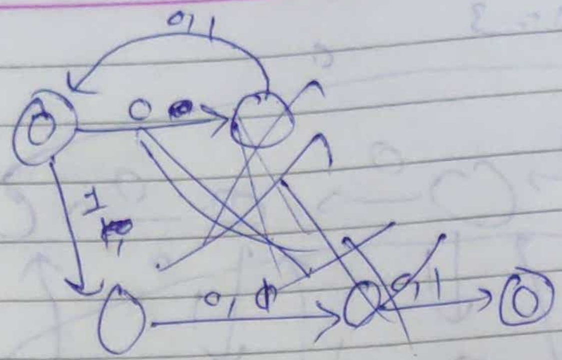
no. of state in DFA for the language over the alphabets {0,1} in which no. 0's are divisible m & no 1's are divisible n is m x n

* for strings (0,1) in which length of string are either divisible by m or divisible n

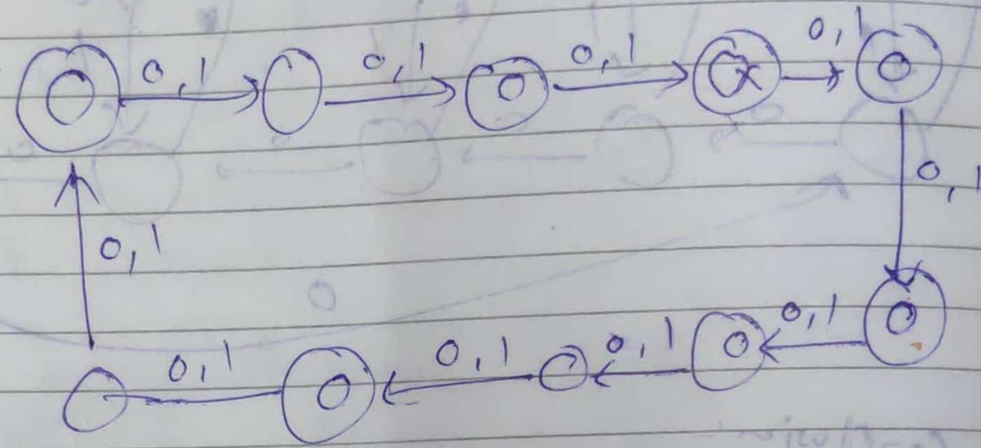
m = 2 n = 5

$$[(0+1)(0+1)]^* + [(0+1)^5]^*$$

0 2 4 6 8



111111

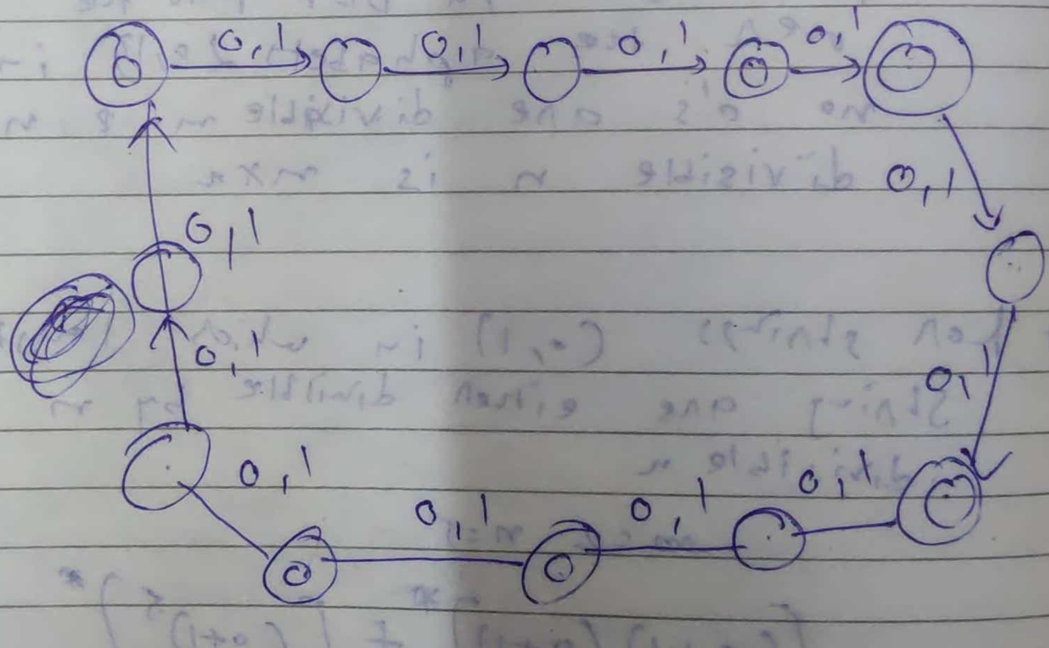


m=3 n=4

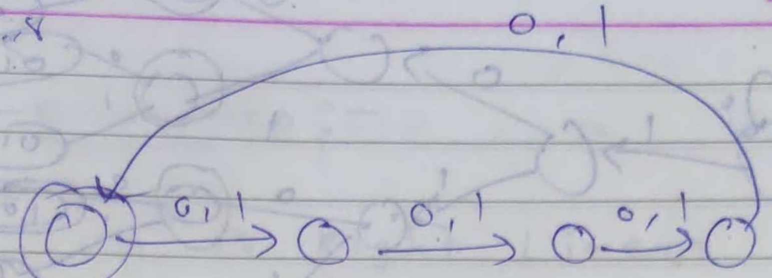
0 3 4

3 6 9 8

4×3



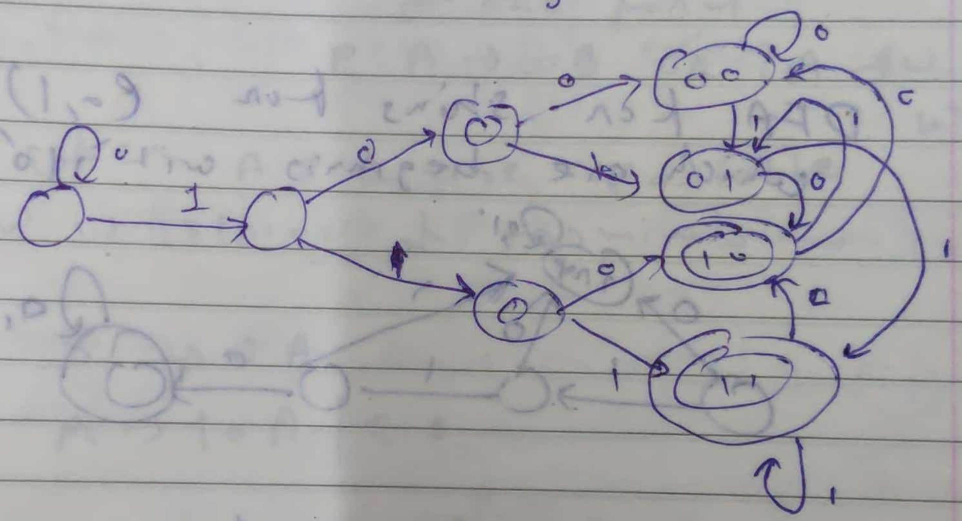
= $m=4$ mod



Conclusion

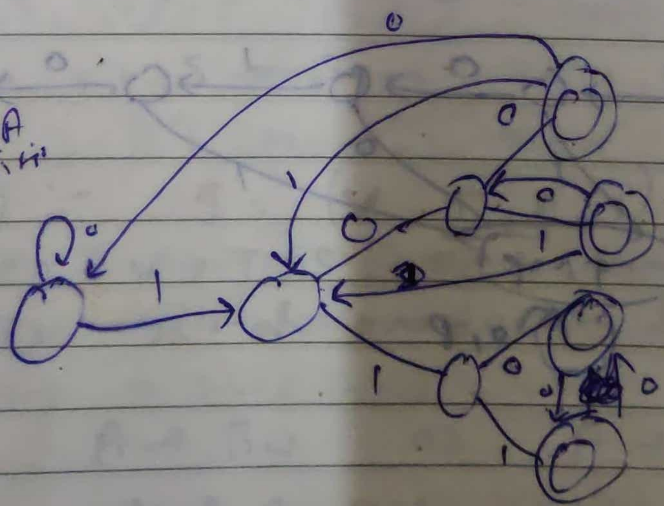
no. of state L.C.M(n,m) ≠ m
and L.C.M.(n,m) ≠ n otherwise
it is MIN(n,m)

* DFA {0,1} is contain 1 in second
position from the right

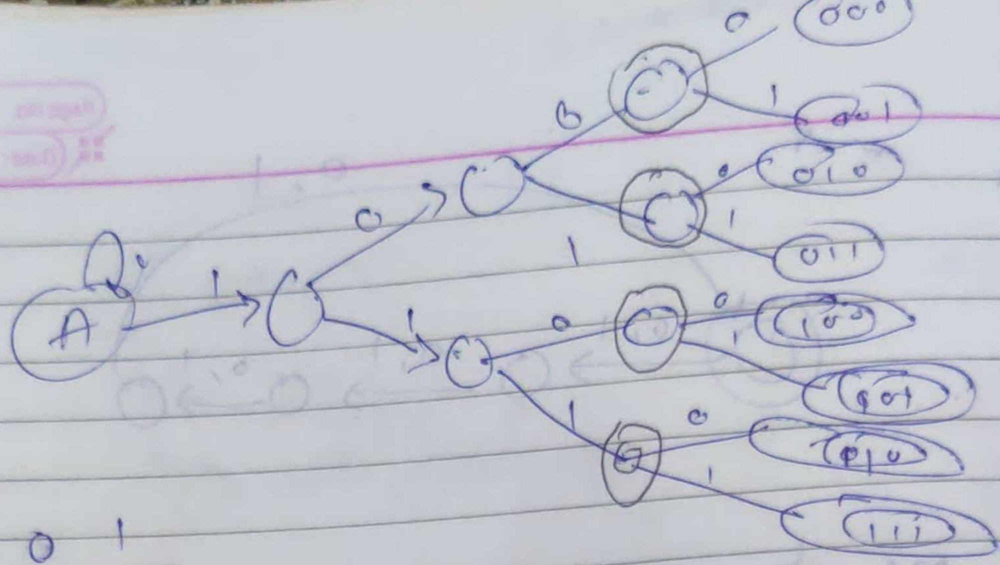


	0	1
0	0	1
1	2	3
2	0	1
3	2	3

* DFA
in 3rd position

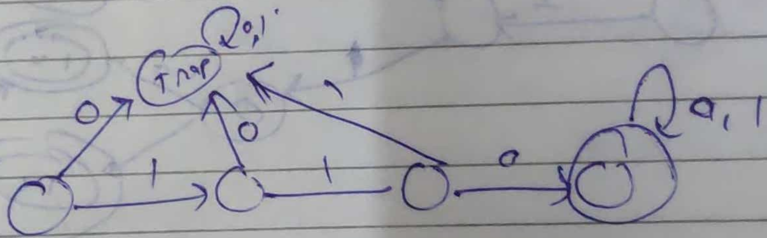


11000

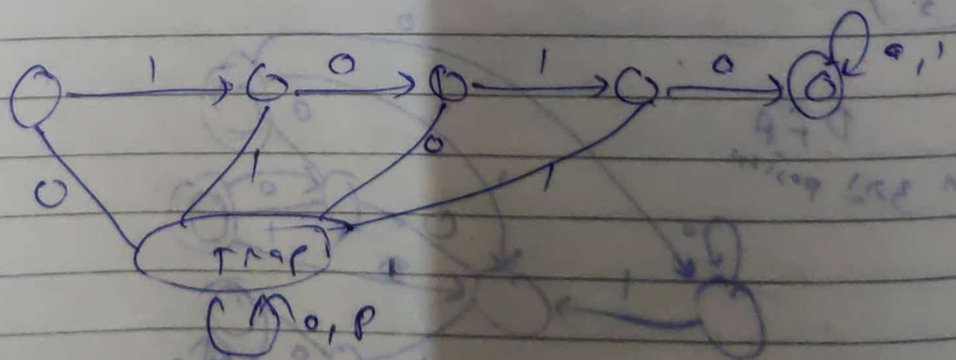


	0	1
0	0	1
1	2	3
2	4	5
3	6	7
4	0	1
5	2	3
6	4	5
7	6	7

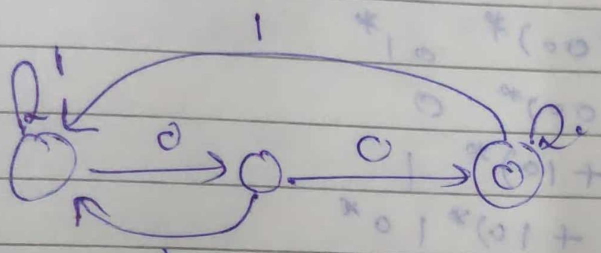
* DFA for string for (0,1) which are beginning with '110'



* beginning with '1010'



which one of following is wrong



above DFA accepts set of all string over (0,1) that

a) begin with either '0' or '1' 01

b) end with 0 10

c) end with 00

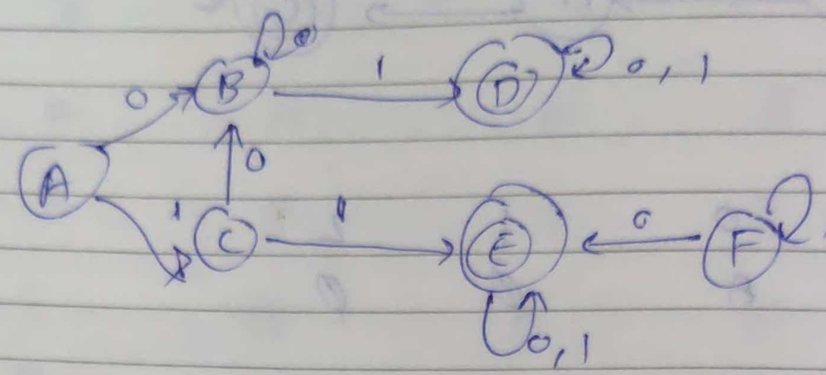
d) containing the substring '00' 00

* The minimum state DFA accepting the language $L = \{ \omega \mid \omega \in \{0,1\}^* \}$ no. of 0's and 1's in ω are divisible by 3 and 5 respectively has

1) 15 state　　3) 10 state

2) 11 state　　4) 9 state

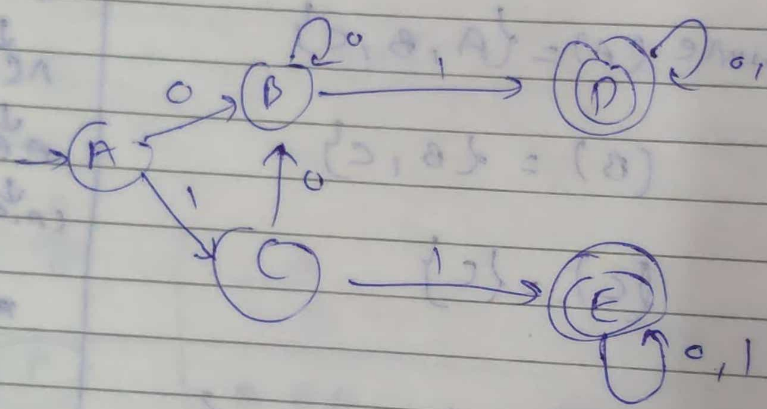
* Minimization of DFA :-



step 1 : remove unreachable state

F

start state થી
કારણ કે કોઈ પાથ નથી ક્યા આવે છે..



	A	B	C	D
B	X			
C	X			
D	X	X	X	
E	X	X	X	

Step - 2 Consider pair of state (q, q) in which one final state, other is non final state and then mark it (x)

(A, D) (B, D) (C, D) (A, E) (B, E) (C, E)

2(b) for the pair of the form (p, q) compute $\delta(p, q) \xrightarrow{a} (r, s)$

if (r, s) is already marked then mark (p, q)

$\delta(AB) \xrightarrow{0} (B, B)$
$\xrightarrow{1} (C, D)$

$\delta(A\ C) \xrightarrow{0} (B, B)$
$\xrightarrow{1} (C, E)$

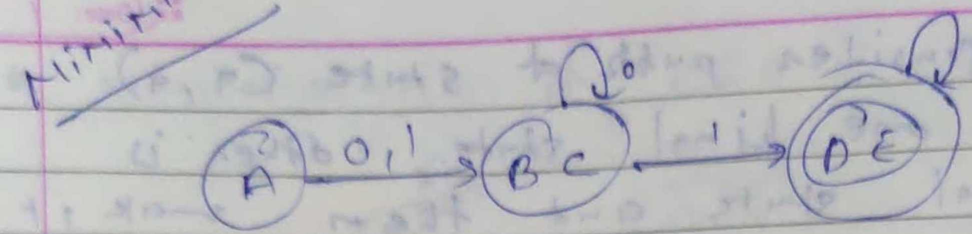
$\delta(B, C) \xrightarrow{0} (B, B)$
$\xrightarrow{1} (D, E)$

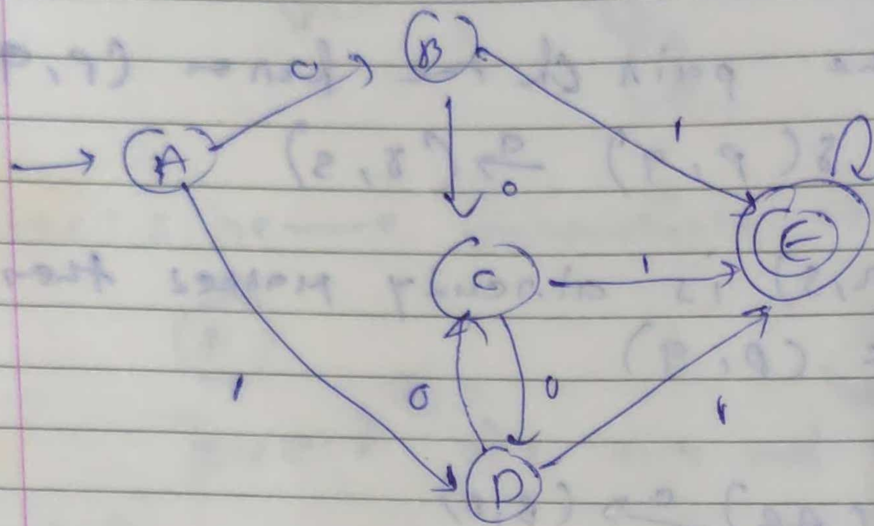
$\delta(D, E) \xrightarrow{0} (D, E)$
$\rightarrow (D, E)$

$\boxed{B = C}$

$\boxed{D = E}$

Minimize?





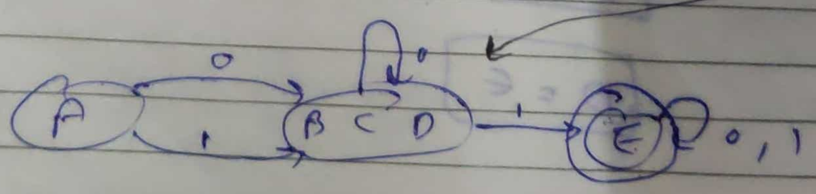
B	X			
C	X			
D	X			
E	X	X	X	X
	A	B	C	D

	A	B	C	D	E
0	B	C	D	C	E
1	D	E	E	E	E

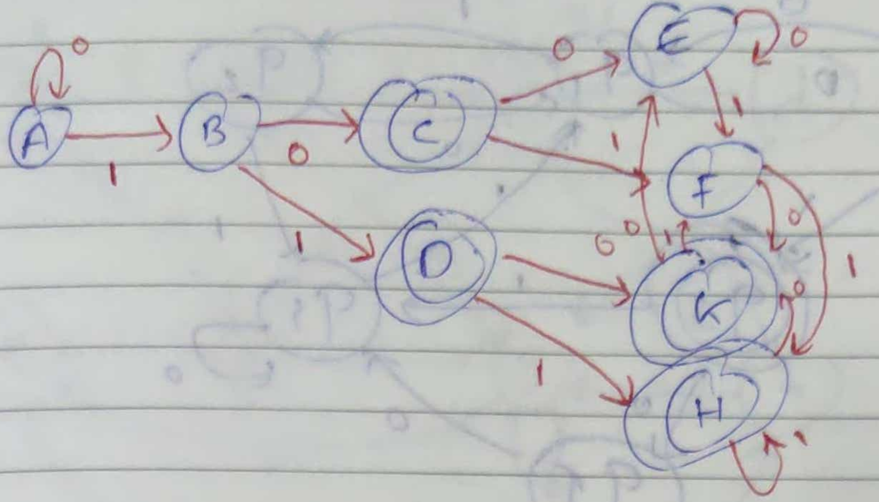
B = C C = D

B = D

BCD



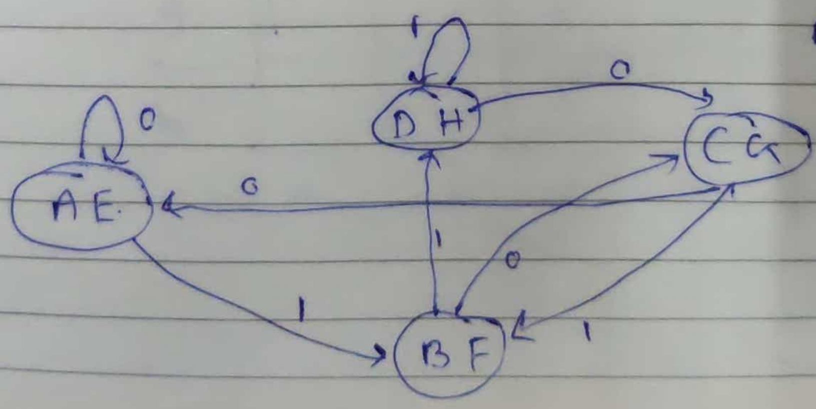
A	B	C	D	E	F	G	H
1	1	1	1	1	1	1	1
B	D	F	H	F	H	F	H



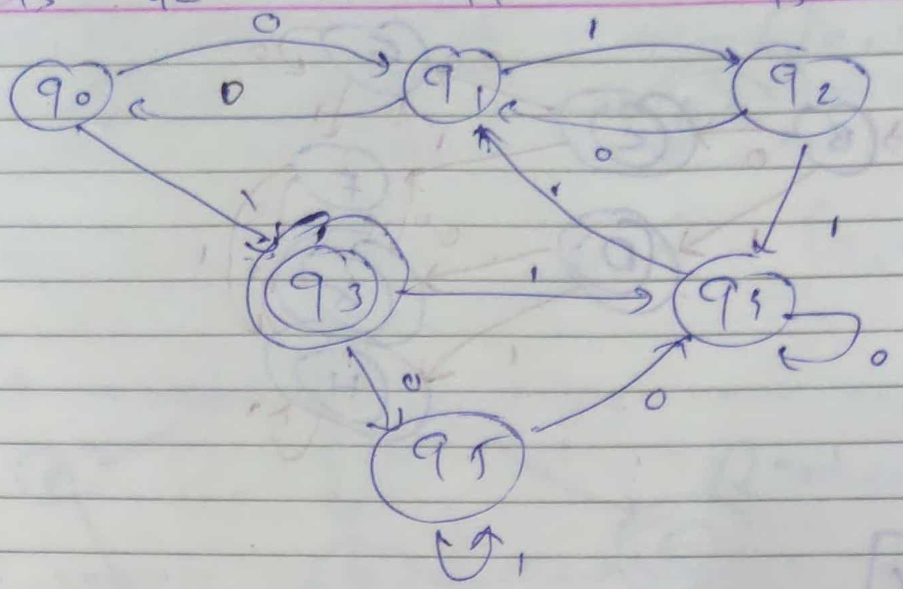
	A	B	C	D	E	F	G
B	X						
C	X	X					
D	X	X	X				
E		X	X	X			
F	X		X	X	X		
G	X	X		X	X	X	
H	X	X	X		X	X	X

D = H = C = G

B = F   A = E

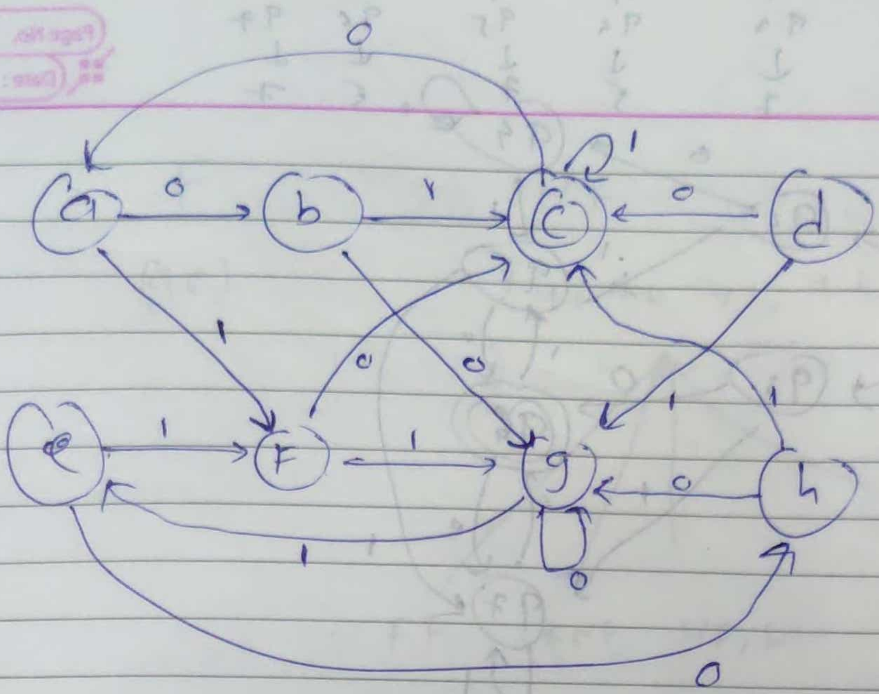


q_1	q_0	q_1	q_5	q_4	q_4
$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$
q_0	q_1	q_2	q_3	q_4	q_5
$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
q_3	q_2	q_4	q_4	q_1	q_5



q_1	X				
q_2	X	X			
q_3	X	X	X		
q_4	X	X	X	X	
q_5	X	X	X	X	X
	q_0	q_1	q_2	q_3	q_4

given DFA is itself minimized



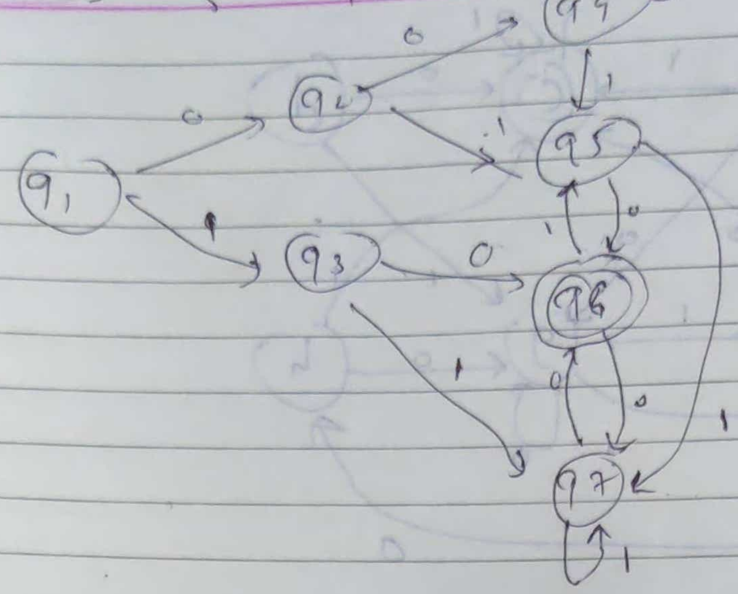
b	g	a		h	c	g	g
↑	↑	↑		↑	↑	↑	↑
a	b	c	~~X~~	E	F	G	H
↓	↓	↓		↓	↓	↓	↓
F	c	c		F	g	e	c

b						
c	X	X				
e			X			
F			X			
g			X			
h			X			
	a	b	c	e	F	g

a = e
b = h

	q_1	q_2	q_3	q_4	q_5	q_6	q_7
↑	2	4	6	4	6	7	6
↓	3	5	7	5	7	5	7

*

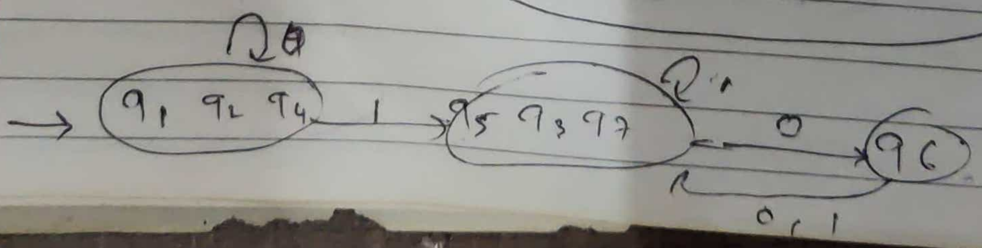


q_2	.					
q_3	X	X				
q_4	.	.	X			
q_5	X	X	.	X		
q_6	X	X	X	X	X	
q_7	X	X	.	X	.	X
	q_1	q_2	q_3	q_4	q_5	q_6

5 = 7   3 = 7

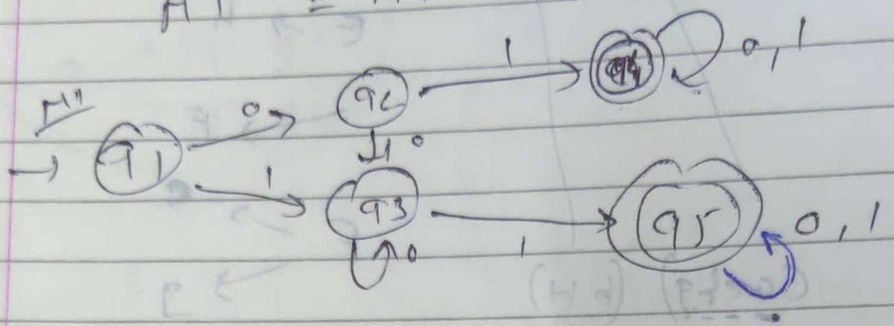
3 = 5

2 = 4   1 = 4   1 = 2

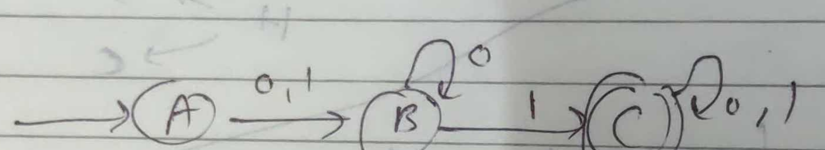


* Equivalence of two DFA

M1 = M1



→ M2



પહેલા સ્ટાર્ટીંગ સ્ટેટ ની સ્ટાર્ટ કરવાનું અને
~~પેર~~ પેર કરવાનું like

$(q_1, A) \rightarrow ((q,0), (A,0)$

જો (F,F)

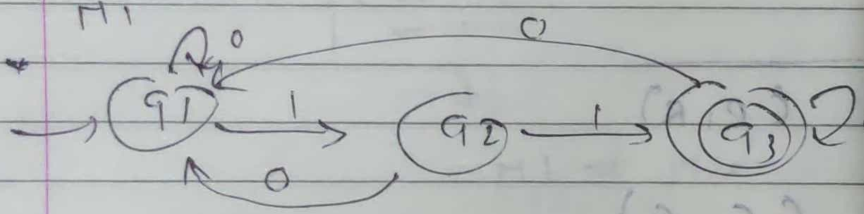
અને (NF NF) હોય તો ઠીક
આગળ ચાલવાનું જ્યાં સુધી નવી પેર આવતી
રહે અને ક્યાંય when નવી પેર ના આવે.

જો સ્ટેટ (F, NF) ~~અથવા~~ or (NF, F) મળે તો
સ્ટોપ થઈ જાય એટલે આ બંને
Equivalent નથી હોય.

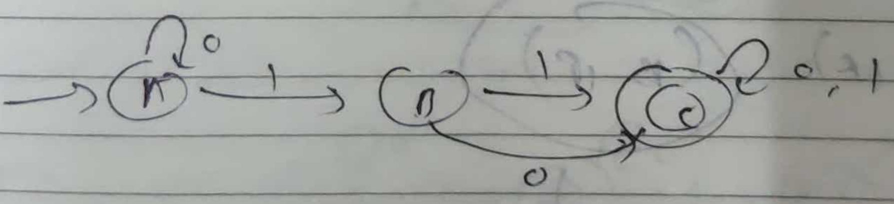
	0	1
(q_1, A)	(q_2, B)	(q_3, B)
(q_2, B)	(q_3, B)	(q_5, C)
(q_3, B)	$(q_3, B$	(q_5, C)
(q_4, C)	(q_4, C)	(q_5, C)
(q_5, C)	(q_5, C)	(q_5, C)

$M_1 \equiv M_2$

M1



M2



	0	1
(q1, A)	(q1, A)	(q2, B)
(q2, B)	(q1, C)	

ત્યારે final
છે ત્યારે non final
નથી So.

M1 ≠ M2

M1

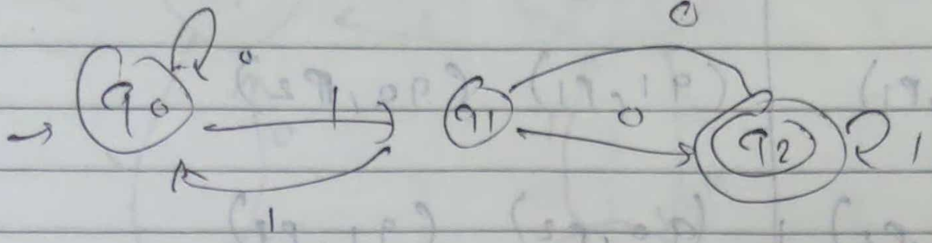
	0	1
→A	B	D
B	A	C
C	D	B
D	C	A

	0	1
→P	R	R
Q	R	P
R	P	Q

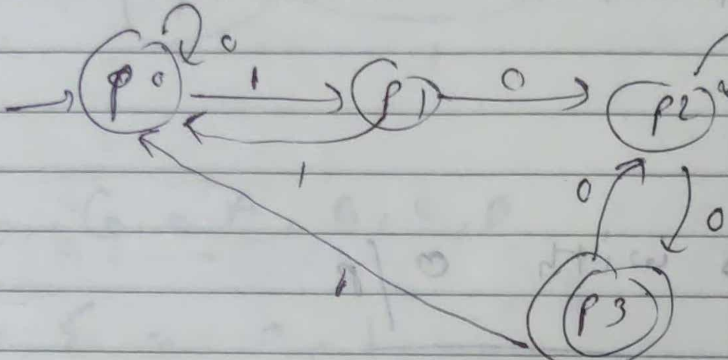
(A, P)	(B, R)	(D, R)
(B, R)	(A, P)	(C, Q)
(D, R)	(C, P)	(A, Q)
(C, Q)	(D, R)	(B, P)

B is final in M1
where P is not
So M1 ≠ M2

M1



M2

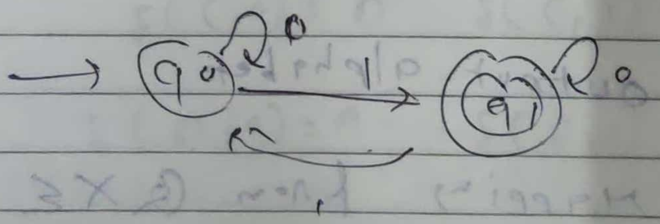


બે સ્ટેટને final કરતા તો $M1 = M2$

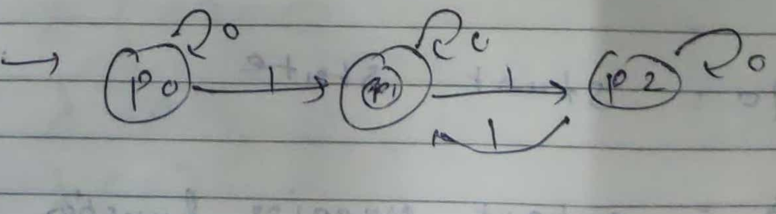
	0	1
(q_0, p_0)	(q_0, p_0)	(q_1, p_1)
(q_1, p_1)	(q_2, p_2)	

$M1 \neq M2$

* M1



M2



	0	1
(q_0, p_0)	(q_0, p_0)	(q_1, p_1)
(q_1, p_1)	(q_1, p_1)	(q_0, p_2)
(q_0, p_2)	(q_0, p_2)	(q_1, p_1)

$M_1 = M_2$

* NFA → DFA

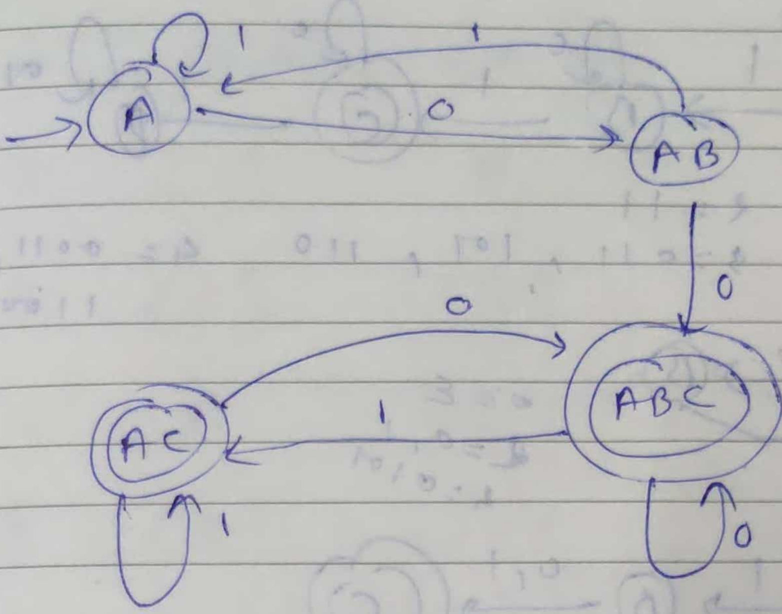
step 1 → table representation

step 2

final state

δ	0	1
→ [A]	[AB]	[A]
[AB]	[ABC]	[A]
[ABC]	[ABC]	[AC]
[AC]	[ABC]	[AC]

becoz in NFA
c is final state
in new state of DFA
where it is c
that will be
final state in DFA



Length that accepted

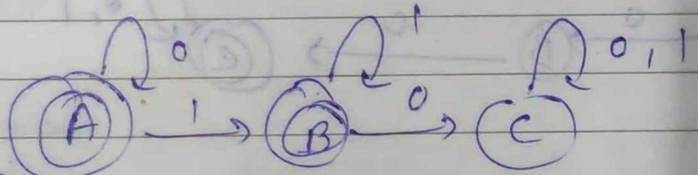
2 = 00

3 = 000, 100, 001

4 → 0000, 0100, 1000, 1100, 1001, 0001, 0010, 0011

$\boxed{\text{substring as 00}}$

(1)



ans: some 0 follow by some 1 — must be ε

0 = ε

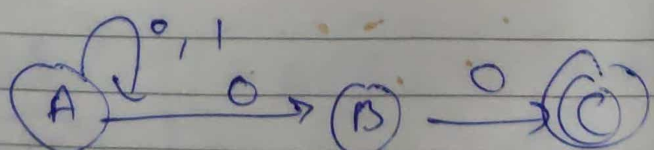
1 → 0, 1

2 → 00, 01, 11

3 → 000, 001, 011, 111

(2)

ans: end with 00



2 = 00

3 = 000

= 100

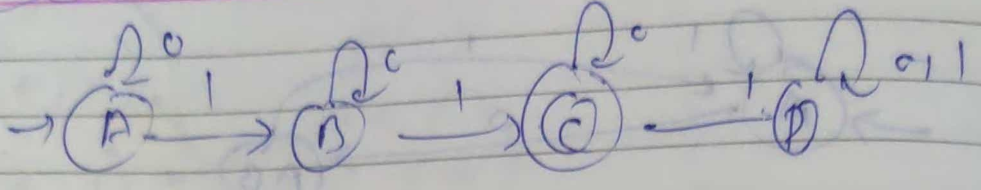
4 = 0000

0100

1000

1100

exactly two 1's



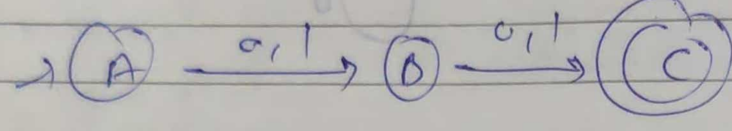
2 = 11

3 = 011, 101, 110   4 = 0011, 0101, 01
1100

0 = ε

2 = 0,1

4 = 0101

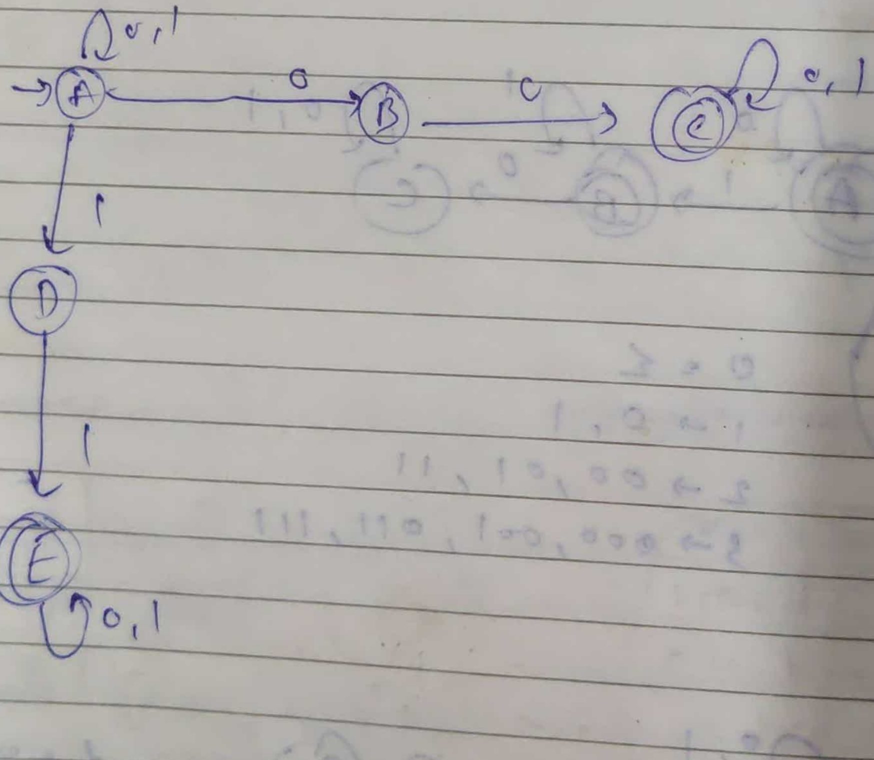


length = 2

2 = 00
01
10
11

Exactly

Convert following NFA to DFA



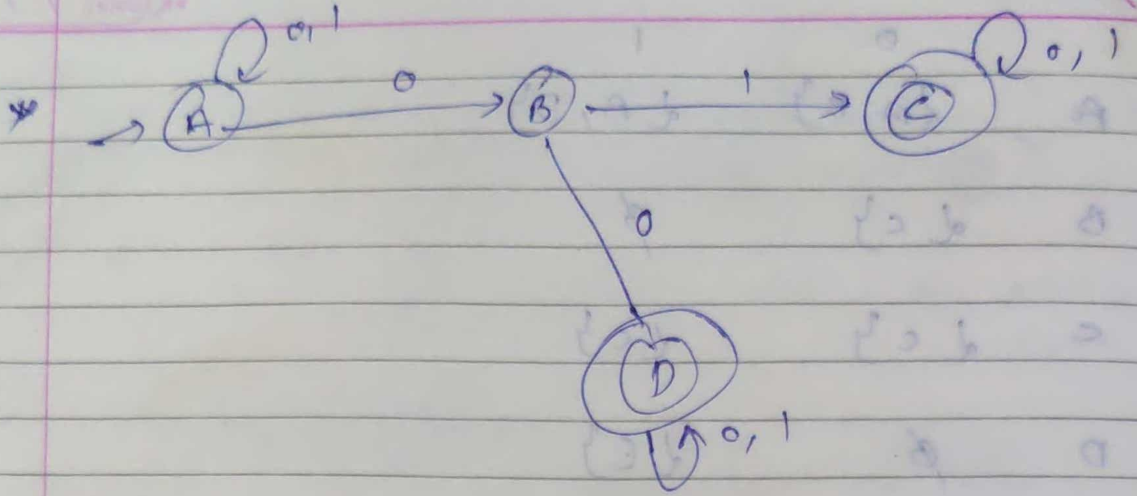
	0	1
A	{A, B}	{A, D}
B	{C}	ϕ
C	{C}	{C}
D	ϕ	{E}
E	{E}	{E}

starting

final state

final state

	0	1
→ A	[A, B]	[A, D]
A, B	[A B C]	[A, D]
A D	[A B]	[A D E]
ABC	[ABC]	[A D C]
ADE	[A B E]	[A D E]
ADC	[ABC]	[A D C E]
ABE	[A B C E]	[A D E]
ADCE	[A B C E]	[A D C E]
ABCE	[A B C E]	[A D C E]



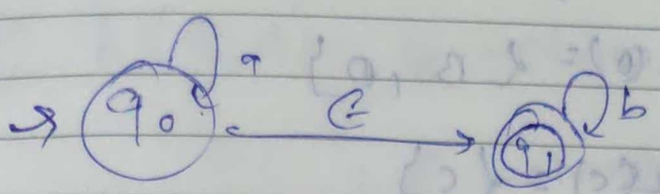
NFA

	0	1
A	[A B]	[A]
B	[D]	[C]
C	[C]	[C]
D	[D]	[D]

DFA

δ	0	1
[A]	[AB]	[A]
[AB]	[ABD]	[AC]
ABD	[ABD]	[ACD]
AC	[AB C]	[AC]
[ACD]	[ABCD]	[ACD]
[ABC]	[ABCD]	[AC]
[ABCD]	[A BCD]	[ACD]

NFA with ϵ moves



$\{\epsilon, a, b, ab, aa, abb,$

* 5 - Tuples

$M = (Q, \Sigma, \delta, q_0, F)$

Excep δ remaining 4 same as DFA

δ is mapping from $Q \times \Sigma \cup \{\epsilon\} \to 2^Q$

$Q \times (\Sigma \cup \{\epsilon\}) \to 2^Q$

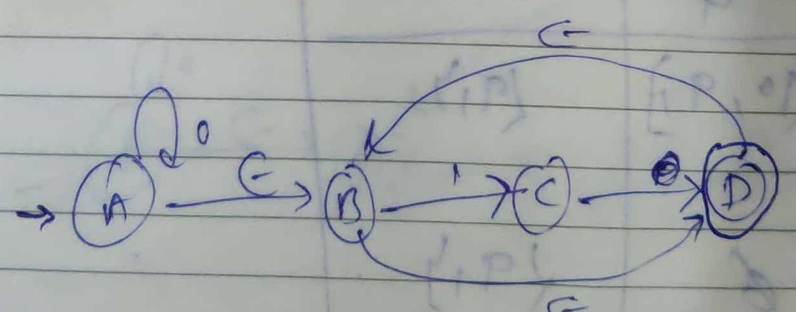
ϵ - closure (q)

set of all state that are reachable from state q. with ϵ moves only.

ϵ closure $(q_0) = \{q_0, q_1\}$

ϵ - closure $(q_1) = \{q_1\}$

*



ϵ closure(A) = {A, B, D}

ϵ closure(B) = {B, D}

ϵ closure(C) = {C}

ϵ closure(D) = {D, B}

* Conversion of NFA - $\wedge$ to NFA

step 1 :-

first rule

	a	b	ϵ
$\rightarrow q_0$	q_0	ϕ	q_1
q_1	ϕ	q_1	ϕ

Step 2 Compute ϵ close for each state.

ϵ closure(q_0) = {q_0, q_1}

" (q_1) = {q_1}

$\hat{\delta}$	a	b
$\rightarrow q_0$	{q_0, q_1}	{q_1}
q_1	ϕ	{q_1}

① $\hat{\delta}(q, \epsilon) = \epsilon\text{-closure}(q)$

② $\hat{\delta}(q, a) = \epsilon\text{-closure}(\delta(\underline{\hat{\delta}(q, \epsilon)}, a))$

$= \epsilon\text{-closure}(\delta(\underline{\{q_0, q_1\}}, a))$

$(q_0, a) \cup (q_1, a)$

$\epsilon\text{-clouse}(q_0)$

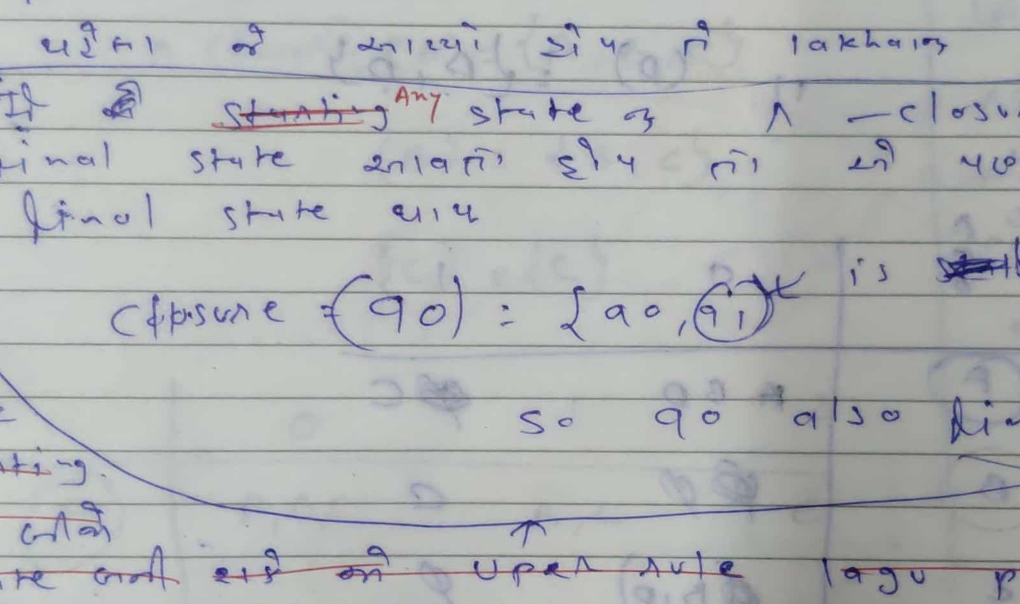
$= q_0, q_1$

* final state માટે
→ પહેલા જે આવ્યો હોય તે lakhavo
→ If કોઈ ~~starting~~ Any state ના Λ-closure માં final state આવતો હોય તો તે પણ final state થાય

closure $(q_0) = \{q_0, q_1\}$ ← is ~~st~~ final state

so q_0 also final state.

~~final state~~
~~sivay starting~~
~~state ne jo~~
~~final state hoi sake ane upar rule~~ lagu padeto.



δ	0	1	ϵ
A	A	ϕ	B
B	ϕ	C	D
C	D	ϕ	ϕ
D	ϕ	ϕ	B

ϵ closure (A) = {A, B, D}

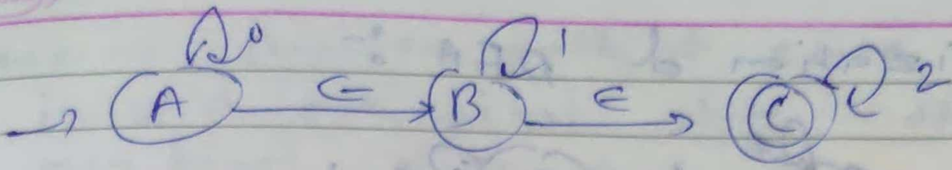
ϵ closure (B) = {B, D}

" (C) = {C}

" (D) = {B, D}

$\hat{\delta}$	0	1
→ A	A B D	$\cancel{\phi}$ C
B	$\cancel{\phi}$ ϕ	C
C	{B, D}	ϕ
D	ϕ	C

as ϵ-closure (B) contain final state



	0	1	2
A	A	∅	∅
B	∅	B	∅
C	∅	∅	C

A closure (A) = {A, B, C}

(B) = {B, C}

{C} = {C}

	0	1	2
→ A	A, B, C	B, C	C
B	∅	B C	C
C	∅	∅	C

(A, 0) M·-2
↓
∧(A)
↓
A B, C
↓
(A, 0), (B, 0) (C, 0)
↓
~~∧~~(A)
↓
∧(A)
↓
$\boxed{ABC}$

Q) ε - closure(B)
contain final state.

* FA → with O/P

Moore Machine

* $M = \{Q, \Sigma, \Delta, \delta, q_0, \lambda\}$

Q = finite set of state

Σ = input alphabet

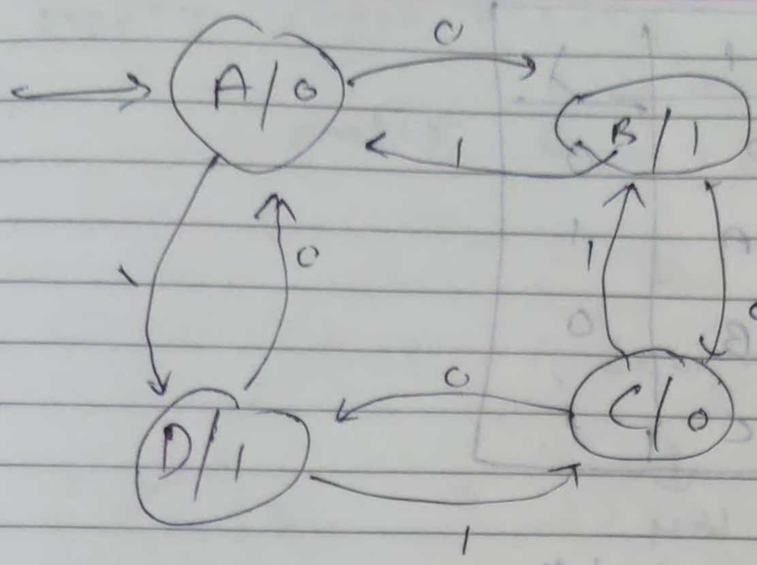
Δ = output alphabet

δ → Mapping from $Q \times \Sigma \rightarrow Q$

q_0 : start state

λ = output mapping function

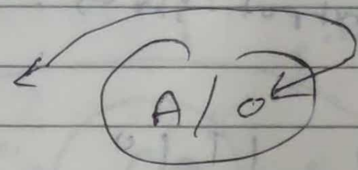
$\lambda : Q \rightarrow \Delta$



$Q = A, B, C, D$

$\Sigma = 0, 1$

$\Delta = \{0, 1\}$



$q_0 = A$

$\delta = \delta(A, 0) = B \quad \delta(A, 1) = D$

$\delta(B, 0) = C \quad \delta(B, 1) = A$

$\delta(C, 0) = D \quad \delta(C, 1) = B$

$\delta(D, 0) = A \quad \delta(D, 1) = C$

λ :-

$\lambda(A) = 0$

$\lambda(B) = 1$

$\lambda(C) = 0$

$\lambda(D) = 1$

δ	0	1	λ
→ A	B	D	0
B	C	A	1
C	D	B	0
D	A	C	1

Input string 1010

$$\overset{0}{A} \xrightarrow{1} \overset{1}{D} \xrightarrow{0} \overset{0}{A} \xrightarrow{1} \overset{1}{D} \xrightarrow{0} \overset{0}{A}$$

output string: 01010

string 1110

$$\overset{0}{A} \xrightarrow{1} \overset{1}{D} \xrightarrow{1} \overset{0}{C} \xrightarrow{1} \overset{1}{B} \xrightarrow{0} \overset{0}{C}$$

output string = 01010

* Design a Moore M/C for "residue mod 4" operation by taking i/p when each prefix is when a input is taken as equal is decimal value

mod 4 and remainder output aya jaie

suppose input 1010

prefix ∈, 1, 10, 101, 1010

↓

decimal

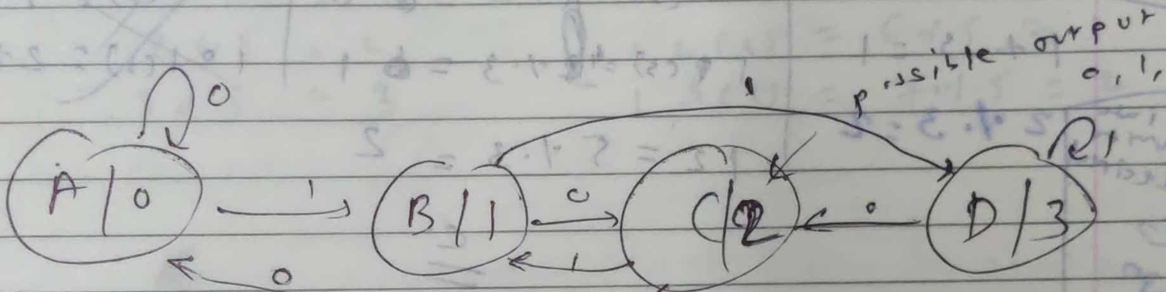
↓

Mod 4

↓

remainder

0 1 2 12



A

i/p out

0 % 4 = 0

1 % 4 = 1

B

10(2) = 2 % 4 = 2

11(2) = 3 % 4 = 3

C

100(2) = 4 % 4 = 0

101(2) = 5 % 4 = 1

minimum path to go to that state B, C & D is 1, 10, 11 respectively

D

110 = 6 % 4 = 2

111 = 7 % 4 = 3

Means at 110 = 6 % 4 = 2

on input 0 go to ~~out~~ that state which output symbol is 2 suppose input 120

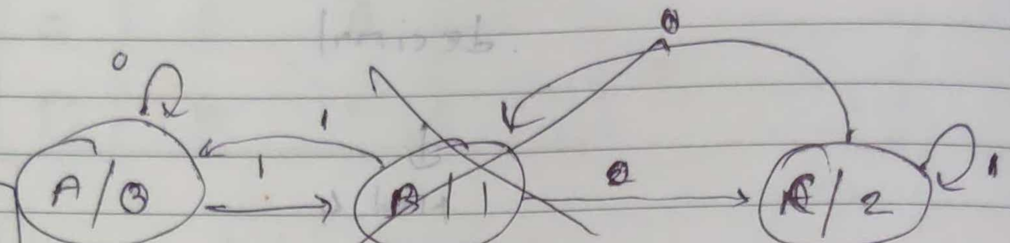
{ ε, 1, 12, 120 }

o/p 0 1 2 0

Q- Same as above

i/p = ternary

"residue mod 3"



11 = 3
$2^1 2^0$
(2×1)+(1×1)
2+1=3

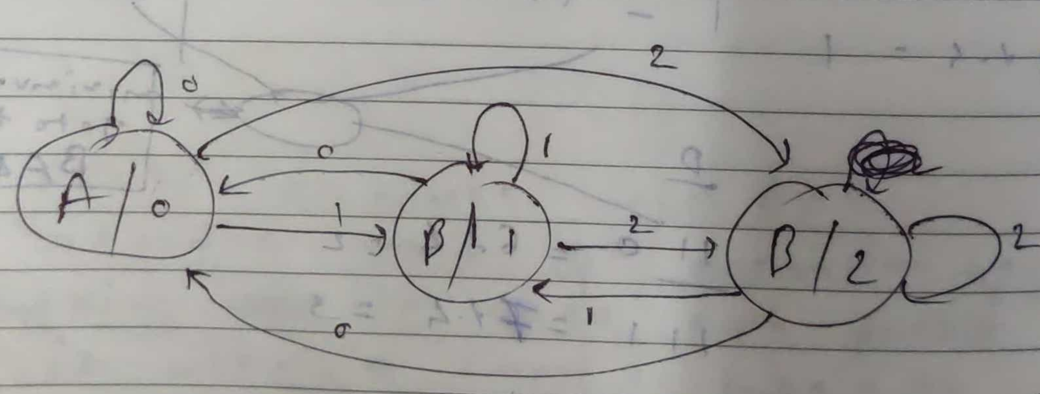
convert into ternary to decimal
20
$3^1 3^0$
6+1×0
6+0=6

A
0 % 3 = 0
1 % 3 = 1
2 % 3 = 2

B
$10_{(3)}$ = 3 % 3 = 0
$11_{(3)}$ = 4 % 3 = 1
12 = 5 % 3 = 2

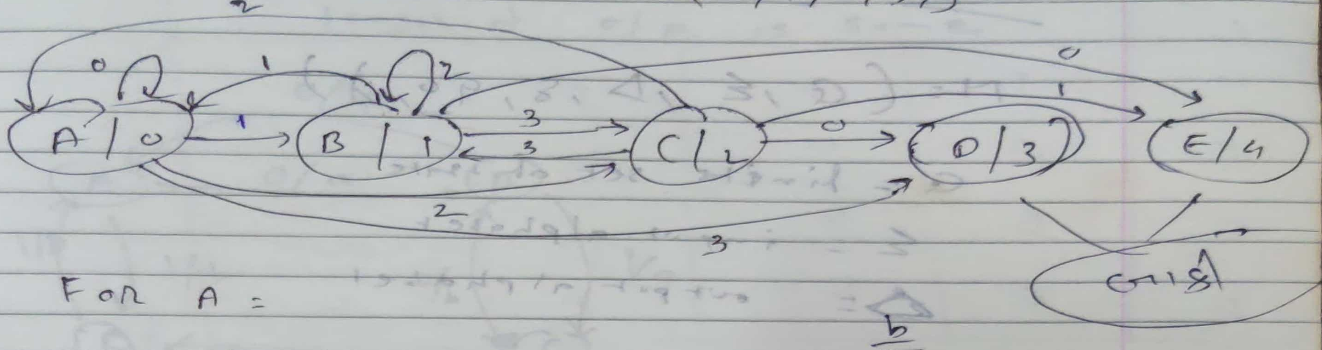
C
~~$100_{(3)}$ = 1 % 3 = 1~~
~~$101_{(3)}$ = 2 % 3 = 2~~

C
$20_{(3)}$ = 6 % 3 = 0
$21_{(3)}$ = 7 % 3 = 1
$22_{(3)}$ = 8 % 3 = 2



Resude mod 5.

input Base = 4 {0, 1, 2, 3, 4}



FOR A =

0 % 5 = 0
1 % 5 = 1
2 % 5 = 2
3 % 5 = 3

B

10(4) = 4 % 5 = 4
11(4) = 5 % 5 = 0
12(4) = 6 % 5 = 1
13(4) = 7 % 5 = 2

C

20 = 8/5 = 3
21 = 9/5 = 4
22 = 10 % 5 = 0
23 = 11 % 5 ≠ 1

D

30 = 12 % 5 = 2
31 = 13 % 5 = 3
32 = 14 + 5 = 4
33 = 15 % 5 = 0

for E

100(4) = 16 % 5 = 1
101(4) = 17 % 5 = 2
102(4) = 18 % 5 = 3
103(4) = 19 % 5 = 4

* Mealy Machine :-

$M = (Q, \Sigma, \Delta, \delta, q_0, \lambda)$

Q = finite set of state

Σ = input alphabet

Δ = output alphabet

$\delta = Q \times \Sigma \rightarrow Q$

q_0 = start state

$\boxed{\lambda = Q \times \Sigma \rightarrow \Delta}$

pitt

→ Case of of mealy machine output depend on state as well as input.

→ In case of moore machine output depend on state only.

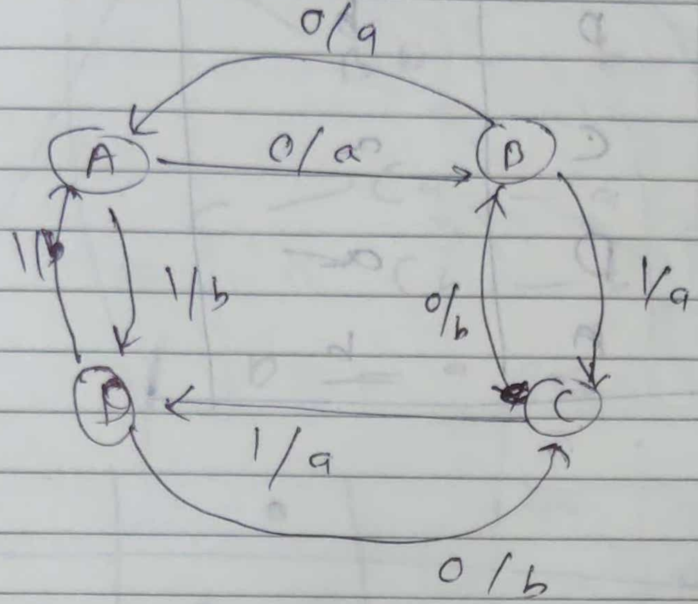
→ with respect of length of i/p and o/p

In case

moore machine, length of o/p is 1 more than length of i/p

Page No.
Date: / /

→ mealy machine length of i/p and length of o/p is same



$Q = \{A, B, C, D\}$

$\Sigma = \{0, 1\}$

$\Delta = \{a, b\}$

δ :

$\delta(A, 0) = B$ $\delta(A, 1) = D$

$\delta(B, 0) = A$ $\delta(B, 1) = C$

$\delta(C, 0) = B$ $\delta(C, 1) = D$

$\delta(D, 0) = C$ $\delta(D, 1) = A$

λ :

$\lambda(A, 0) = a$ $\lambda(A, 1) = b$

$\lambda(B, 0) = a$ $\lambda(B, 1) = a$

$\lambda(C, 0) = b$ $\lambda(C, 1) = a$

$\lambda(D, 0) = b$ $\lambda(D, 1) = b$

δ	0		1	
	N/S	o/p	N/S	o/p
A	B	a	D	b
B	A	a	C	a
C	B	b	D	a
D	C	b	A	b

Input 1011

$$A \xrightarrow{1/b} D \xrightarrow{0/b} C \xrightarrow{1/a} D \xrightarrow{1/b} A$$

* Inter Conversion of Mealy & Moore Machine

Mealy Machine ⟶ Moore Machine

	0		1	
	N/S	o/p	N/S	o/p
A	B	a	D	a
B	A	a	C	a
C	B	b	D	a
D	C	b	A	a

આ લખવા માટે આગળના page ના 0 and 1 ની value check કરવાની. જે state દેખાતા તેમાં વારા વારા o/p હોય તો

$A \to a$

$B \to Ba$ — a

$\quad\;\; Bb$ — b

$C \to Ca$ — a

$\quad\;\; Cb$ — b

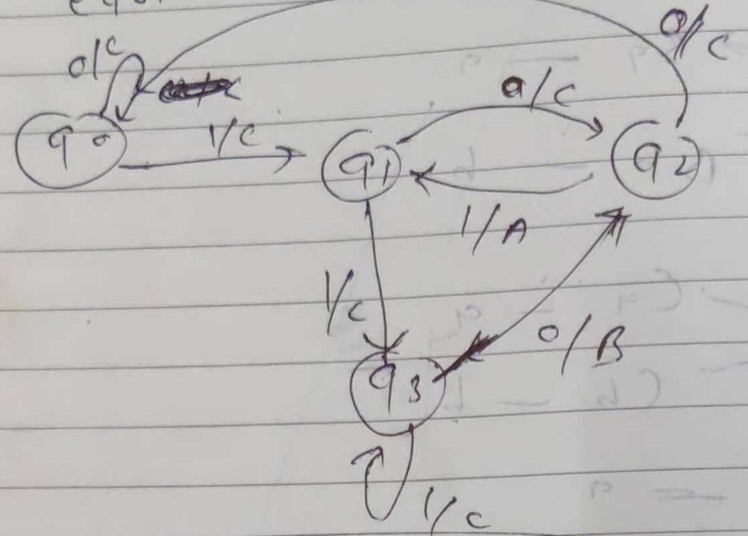
D — a

	0	1	
A	Ba	D	a
Ba	A	Ca	a
Bb	A	Ca	b
Ca	Bb	D	a
Cb	Bb	D	b
D	Cb	A	a

field bharva માટે પહેલા વાળા table ni check કરવાનું like

A, 0 એ આગળના દેખાતા B સ્ટેટ છે પણ આગળ તો Ba and Bb છે so next symbol Jovano નાખું તો પહેલા વાળા દેખાતા જે a છે so ans. Ba

6 Convert the mealy Machine to equivelent Moore Machine



	0		1	
	N.S	O/P	N/S	op
q_0	q_0	C	q_1	C
q_1	q_2	C	q_3	C
q_2	q_0	C	q_1	A
q_3	q_2	B	q_3	C

$q_0 \rightarrow C$

$q_1 \rightarrow q_{1A} \rightarrow A$

$\searrow q_{1C} \rightarrow C$

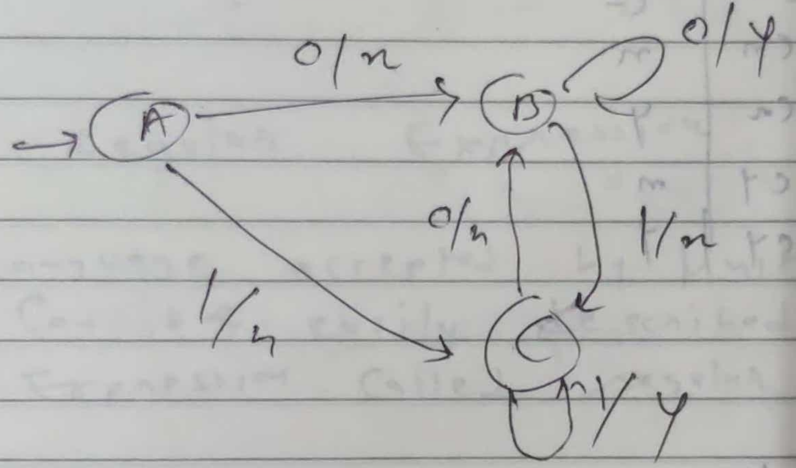
$q_2 \rightarrow q_{2b} \rightarrow B$

$\searrow q_{2C} \rightarrow C$

$q_3 \rightarrow C$

δ	0	1	op
q_0	q_0	q_{1c}	C
q_{1a}	q_{2c}	q_3	A
q_{1c}	q_{2c}	q_3	C
q_{2b}	q_0	q_{1a}	B
q_{2c}	q_0	q_{1a}	C
q_3	q_{2b}	q_3	C

Q. ve



	0		1	
	ns	op	ns	op
A	B	x	C	x
B	B	y	C	x
C	B	x	C	y

B

A

A → ∈ ક્યારેય પણ રેખામાં નથી

B → Bx − x
→ By − y
C → Cx − x
Cy − y

δ	0	1	O/P
→ A	Bx	Cx	∈
Bx	By	Cx	x
By	By	Cx	y
Cx	Bx	Cy	x
Cy	Bx	Cy	y

P Moore to Mealy

Q

δ	0	1	λ
→ A	B	C	a
B	C	D	b
C	A	B	a
D	C	D	b

δ	0		1	
	n/s	o/p	n/s	o/p
A	B	b	C	a
B	C	a	D	b
C	A	a	B	b
D	C	a	D	b

o/p

$A \to a$

$B \to b$

$C \to a$

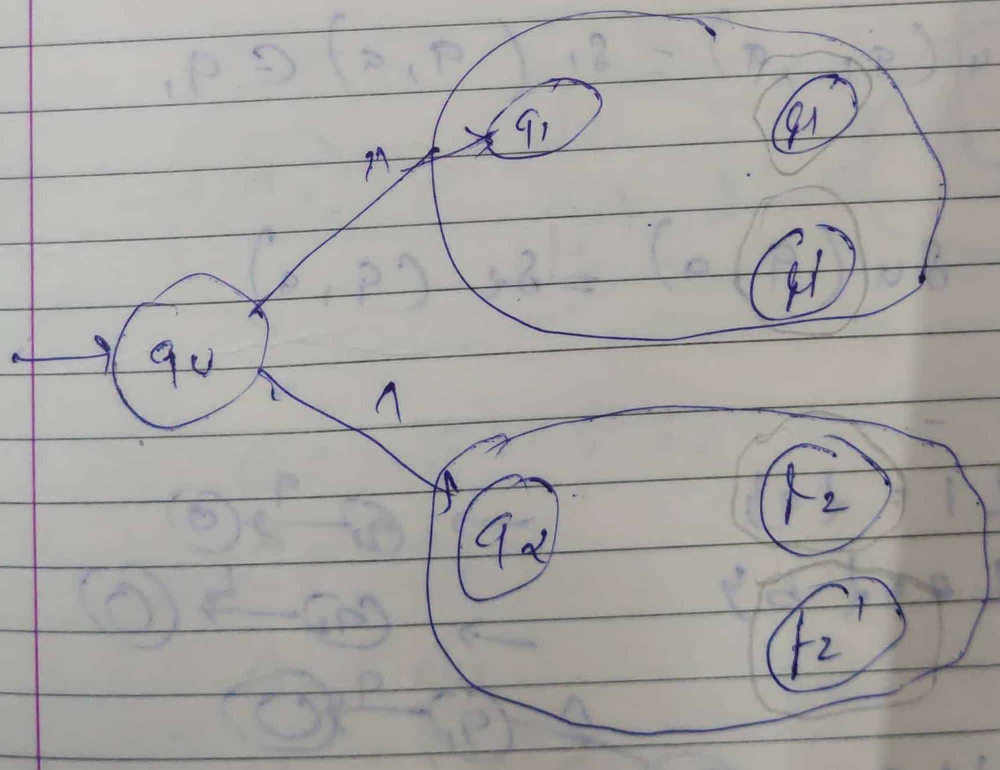
$D \to b$

TOC

Kleene's Therum - I
- Every Regular language accepted by FA

P.R.E
→ $\Lambda \rightarrow \{\Lambda\}$
→ $\phi \rightarrow \{\phi\}$
→ $a \rightarrow \{a\}$
$\in \Sigma$

$r_1 + r_2 \rightarrow \{r_1 \cup r_2\}$
$r_1 r_2 \rightarrow \{R_1 r_2\}$
$r_1^* \rightarrow \{r_1^*\}$



$m_1 = (Q, \Sigma, q_1, A, \delta_1)$

Union : NFA for $L_1 \cup L_2$ is

$M_u = (Q_u, \Sigma, q_u, A_u, \delta_u)$

where

$Q_u = Q_1 \cup Q_2 \cup \{q_u\}$

નો સ્ટાર્ટીંગ સ્ટેટ ઈનીશીયલ
સ્ટા. નવું

$q_u = q_0$

$A_u = A_1 \cup A_2$

$\delta_u(q_u, \Lambda) = \{q_1, q_2\}$

$\delta_u(q_u, a) = \phi$

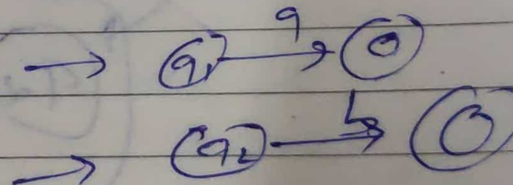
$q \in Q_1$ $\quad \delta_u(q_1, a) = \delta_1(q, a) \in q_1$

$q \in Q_2$ $\quad \delta_u(q, a) = \delta_2(q, a)$

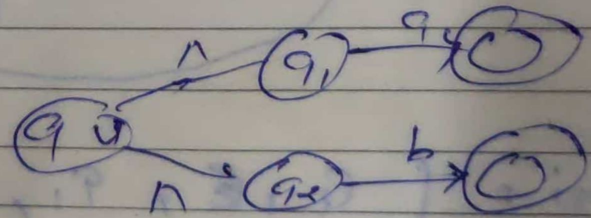
ex

$L_1 = \{a\}$

$L_2 = \{b\}$



$L_1 \cup L_2$

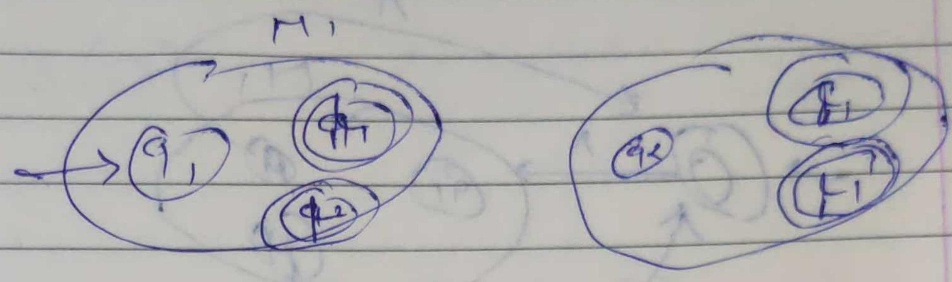


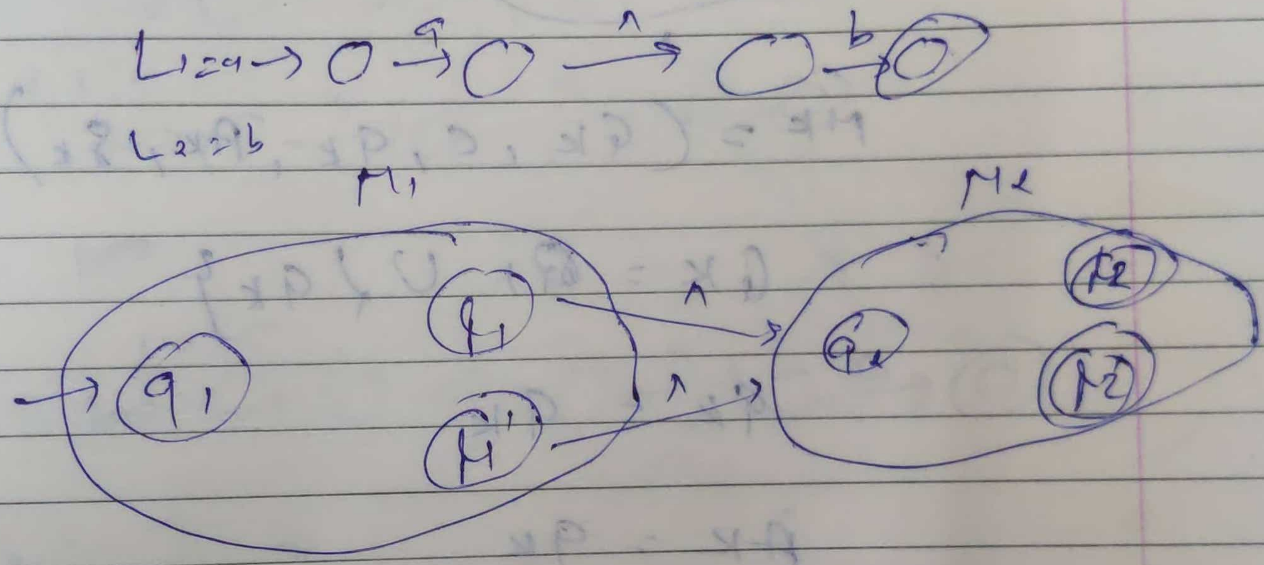
NFA

Concatenation

$L_1 . L_2$

$M_C = (Q_C, \Sigma, q_C, A_C, \delta_C)$





$Q_C = Q_1 \cup Q_2$

$q_C = q_1$

$A_C = A_2$

if $q \in Q_1$

$\delta_C(q, a) = \delta_1(q, a)$

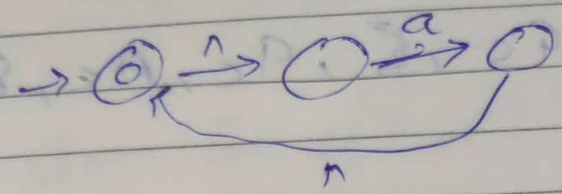
if $q \in A_1$

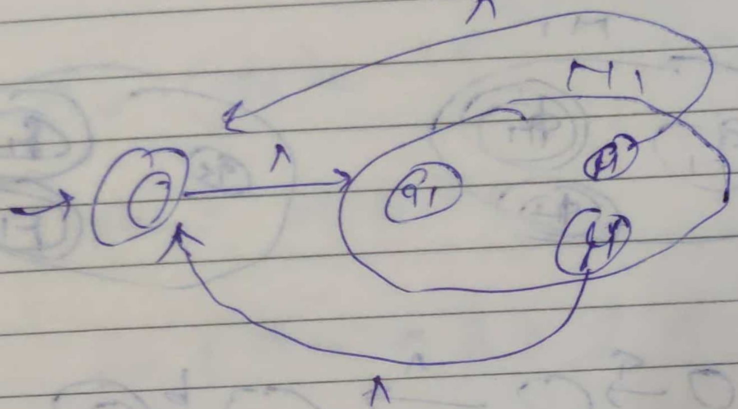
$\delta_C(q, \Lambda) = q_2$

L_1^*

$L_1 = a$

$L_1^* = a^*$





$M_k = (Q_k, \Sigma, q_k, A_k, \delta_k)$

$Q_k = Q_1 \cup \{q_k\}$

$q_k = q_k$

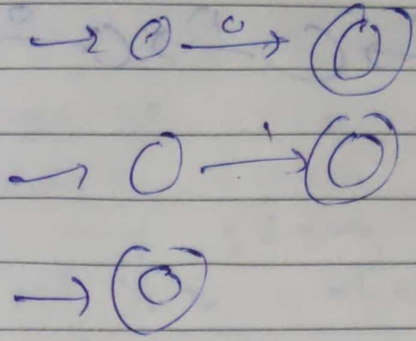
$A_k = q_k$

if $q \in Q_1$. $\delta_k(q, a) = \delta_1(q, a)$

$\delta_k(q_k, \Lambda) \to q_1$

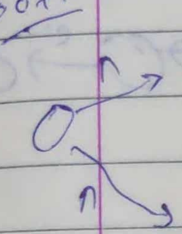
if $q \in A_1$ $\delta_k(q, \Lambda) = q_k$

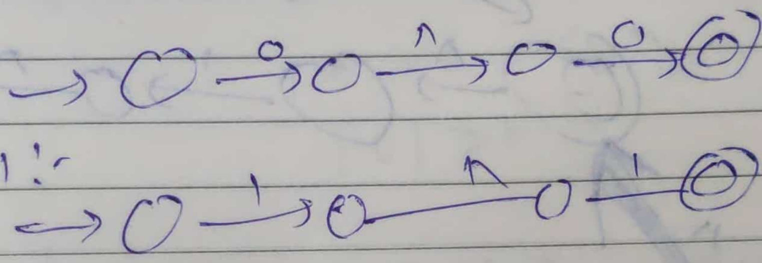
$(00 + 11)^* (10)^*$



00 :-

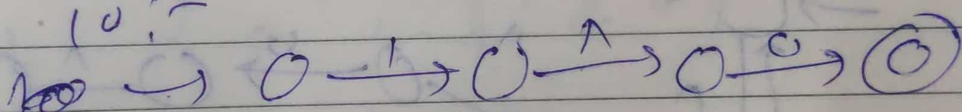
00+11



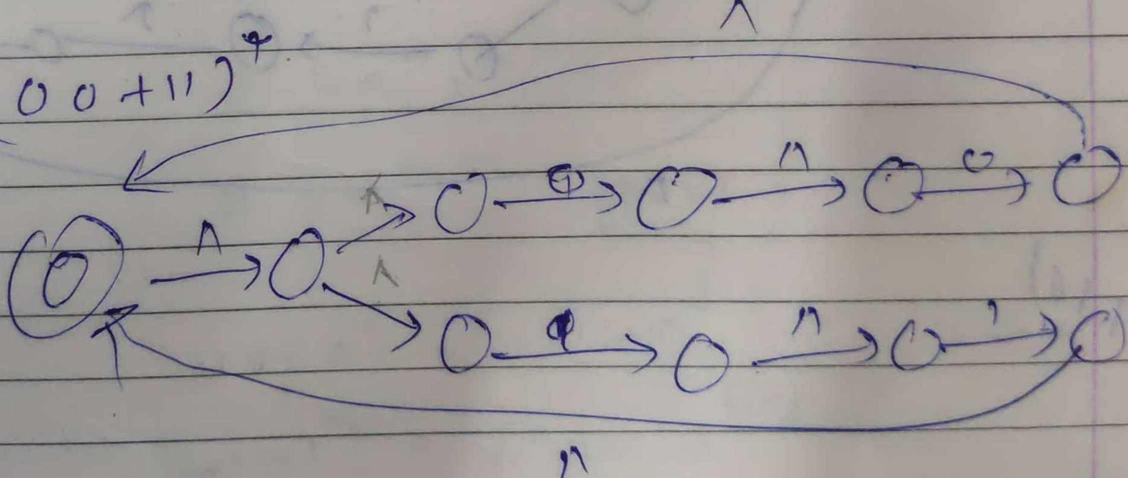


11 :-

10 :-



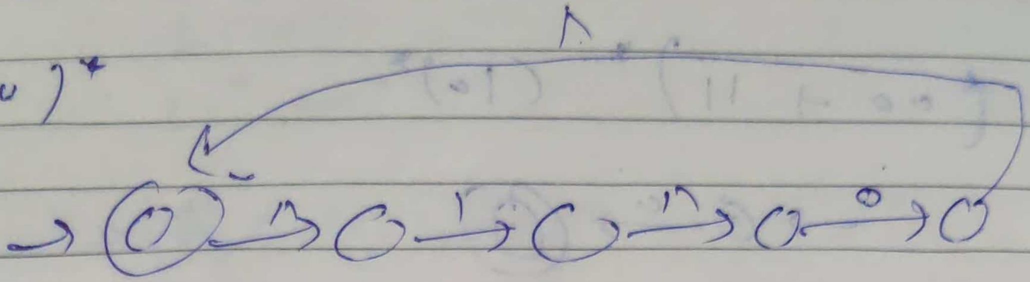
$(00+11)^*$



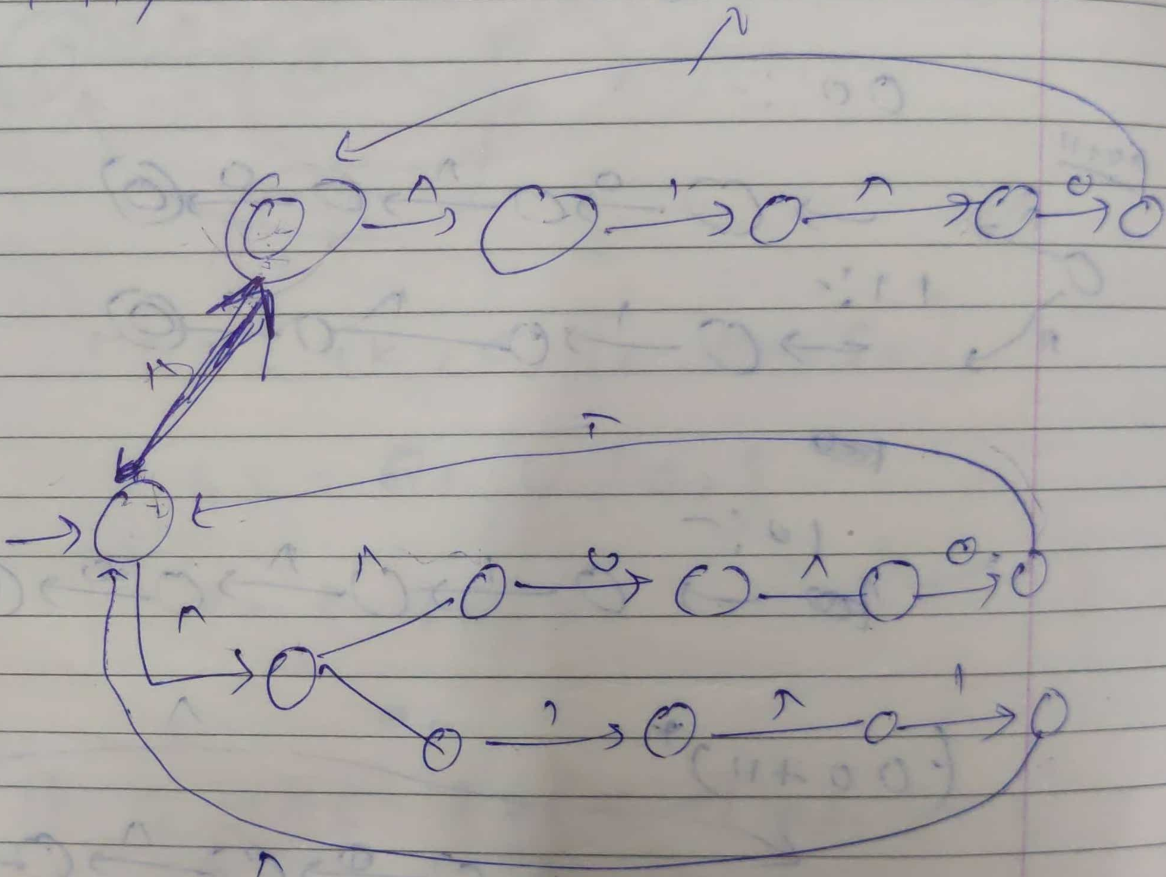
Mid sem xam

$\boxed{1, 2, 3, 4, 5}$

$(10)^*$



$(00+11)^*(10)^*$



(1+0)

* properties of Regular language

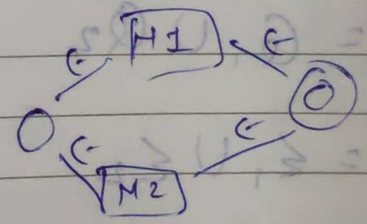
1) regular language are close under following property

1. union
2. concatenation
3. Kleen closure
4. complementation
5) Intersection
6) Reversal
7) substitution
8) Homo-Morphism
9) Inverse Homo-Morphism
10) Init operation

* 1) union

L_1 L_2 $L_1 \cup L_2$

M_1 M_2



$L_1 = M_1 = \{Q_1, \Sigma_1, \delta_1, q_1, F_1\}$

$L_2 : M_2 = (Q_2, \Sigma_2, \delta_2, q_2, F_2)$

$L_1 \cup L_2 : M_3 = \{Q_3, \Sigma_3, \delta_3, q_3, F_3)$

$Q_3 = Q_1 \cup Q_2 \cup \{q_0, f_0\}$

$\Sigma_3 = \Sigma_1 \cup \Sigma_2$ $q_3 = q_0$

$f_3 = f_0$

$\delta_3(q,a) = \delta_1(q,a)$ if $(q \in Q_1)$

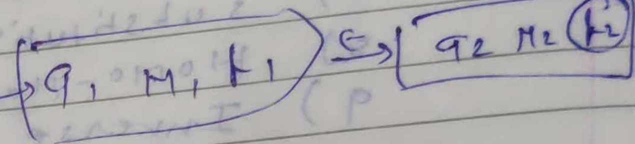
$\delta_3(q,a) = \delta_2(q,a)$ if $(q \in Q_2)$

$\delta_3(q_0, \epsilon) = \{q_1, q_2\}$

$\delta_3(f_1, \epsilon) = f_0 = \delta_3(f_2, \epsilon)$

✓ Regular set are close under concatenation

L_1 | M_1 L_2 | M_2



$L_1 = M_1 = (Q_1$ ——

$L_2 = M_2 = (Q_2$ ——

$L_3 = M_3 = (Q_3$ ——

$Q_3 = Q_1 \cup Q_2$

$\Sigma_3 = \Sigma_1 \cup \Sigma_2$

$q_3 = q_1$

$f_3 = f_2$

$\delta_3 : \delta_3(q,a) = \delta_1(q,a)$ if $(q \in Q_1)$

$\delta_3(q,a) = \delta_2(q,a)$ if $(q \in Q_2)$

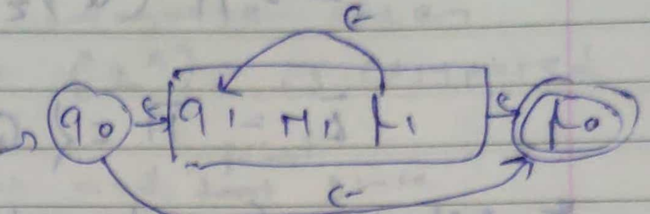
$\delta_3(f_1, \epsilon) = q_2$

* Regular set are closed under kleen closure

L_1

$\downarrow$

M_1



$L_1 = M_1 = (Q_1, \Sigma_1, \delta_1, q_1, F_1)$

$L_1^* = M_3 = (Q_3, \Sigma_3, \delta_3, q_3, F_3)$

$Q_3 = Q_1 \cup \{q_0, f_0\}$

$\Sigma_3 = \Sigma_1$

$q_3 = q_0$

$F_3 = f_0$

$\delta_3 = (q, a) = \delta_1(q, a)$ if $(q \in Q_1)$

$\delta_3 = (q_0, \epsilon) = \{q_1, f_0\}$

$\delta_3 = (F_1, \epsilon) = \{q_1, F_0\}$

* regular set are closed under complement
  1) absolute complement
  2) relative complement

Let L is Regular language for the alphabet $\{0, 1\}$ then absolute complement of L is $\overline{L} = \{0, 1\}^* - L$

L complement of language L can be obtain by interchanging final and non final state of L.

$L = M(Q, \Sigma, \delta, q_0, F)$

$\bar{L}M = \bar{M} = \{Q, \Sigma, \delta, q_0, Q-F)$

* Relative Complement:

If regular language L_1, L_2 is $L_1 - L_2 = \{w \mid w \in L_1$ and $w \in L_2\}$

from $L_1 - L_2 = L_1 \cap \bar{L}_2$

for any given regular language L_2 $\bar{L}_2$ is also regular language.

Hence it is enough to prove regular sets are closed under intersection

let construct L_1 and L_2 two regular language ~~L1 inter~~ $L_1 \cap L_2$ is also regular lang

$$L_1 \cap L_2 = \overline{\bar{L}_1 \cup \bar{L}_2}$$

if L_1 and L_2 are regular language $\bar{L}_1, \bar{L}_2$ are also R.L, Because R.S closed under absolute comple

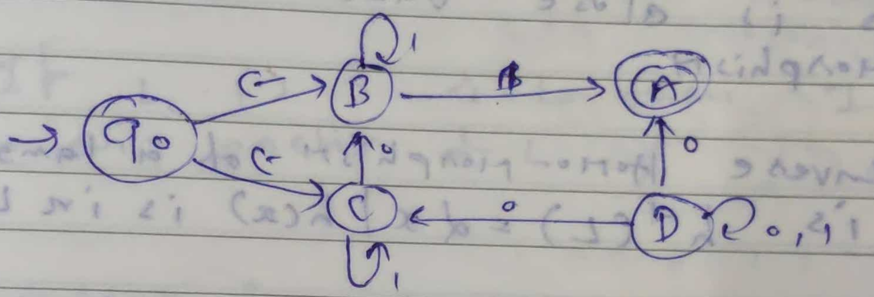
if $\bar{L}_1, \bar{L}_2$ are regular then $\bar{L}_1 \cup \bar{L}_2$ is also regular because R.S are closed under Union.

if $\bar{L}_1 \cup \bar{L}_2$ is regular than $\overline{\bar{L}_1 \cup \bar{L}_2}$ is also a regular.

Hence regular set are closed under intersection

* R.S are close under reversal
if L is regular Language then reversal of L is (L^R) is computed as follow

1) interchange start and final state
2) reverse direction of transition, keeping input as it is.



* R.S close under substitution
In substitution the i/p symbol is replace with a regulare Language

$L = a1 + (11)^*$

$f(a) = a^*$

$f(1) = ab$

$a^* ab + (abab)^*$

* when i/p symbol are replaced with a R.E and result is also a R.E
→ we know Any given R.E there exist equilent finite Automata
→ Hence regular set are closed under substitution

* Homo-Morphism is a special case of substitution where i/p is replace with a string

$$L = 01 + (11)^*$$

$$h(0) = bab$$

$$h(1) = ab$$

$$babab + (abab)^*$$

I/P is replace with string result is also R.E

* R.S is close under Inverse Homo-Morphism

Inverse Homo-Morphism of a language L is $h^{-1}(L) = \{x \mid h(x) \text{ is in } L\}$

$$\Sigma = \{0, 1\} \qquad \Delta = \{a, b\}$$

$$h : \Sigma \to \Delta$$

$$h(0) = aa \qquad h(1) = aba$$

compute

$$L = (ab + ba)^* a$$

$$h^{(-1)}(L) = \{1\}$$

0 ક્યાંય ના આવતો કારણ કે

$h(0) = aa$ છે

L થી answer મળશે

$L = \{a, aba, baa, ababa \ldots\}$

* $\Sigma = \{0, 1, 2\} \qquad \Delta = \{a, b\}$

$h(0) = a \qquad h(1) = ab \qquad h(2) = ba$

Page No.
Date: / /

$L_1 = \{ababa\}$ $L_2 = a(ba)^*$

(1) $h^{-1}(L_1)$ (2) $h^{-1}(L_2)$

$L_1 = \{ababa\}$ $L_2 = \{a, aba, ababa\}$

$h^{-1}(L_1) = \{022, 110, 102\}$ $h^{-1}(L_2) = \{02^*, 1^*0\}$

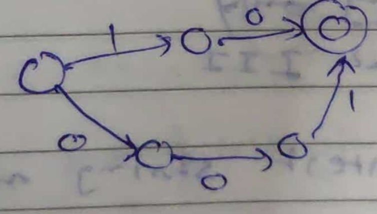
0^* [illegible]

* R.S are close under init operation

If L is a r.L then Init(L) is every prefix of L

The machine for Init(L) can be constructed by making every state of L as final state.

$L = 10 + 001$



operation of language

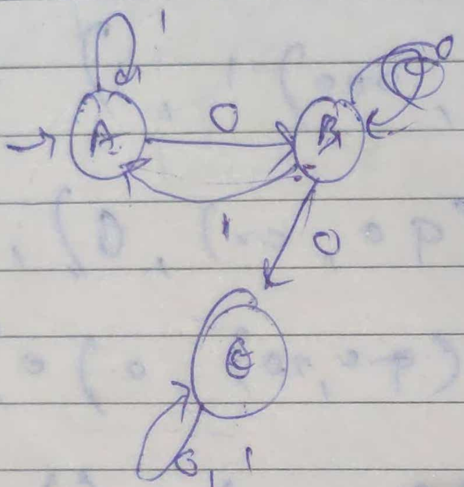
L_1 & L_2 are R.L.

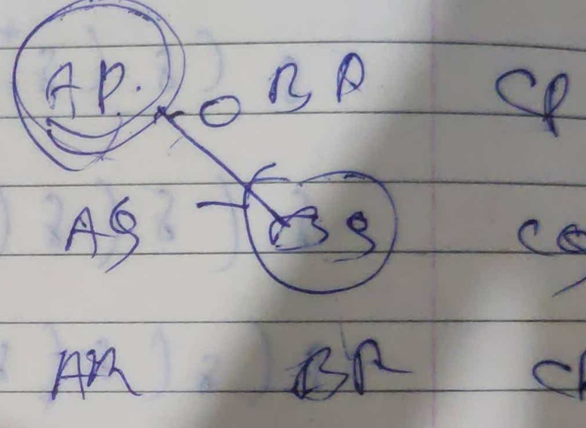
→ $L_1 \cup L_2$

↳ $L_1 \cap L_2$

→ $L_1 - L_2$

contain (00)

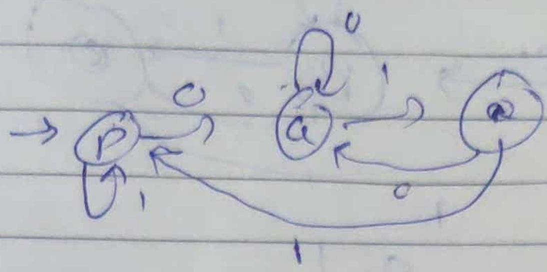




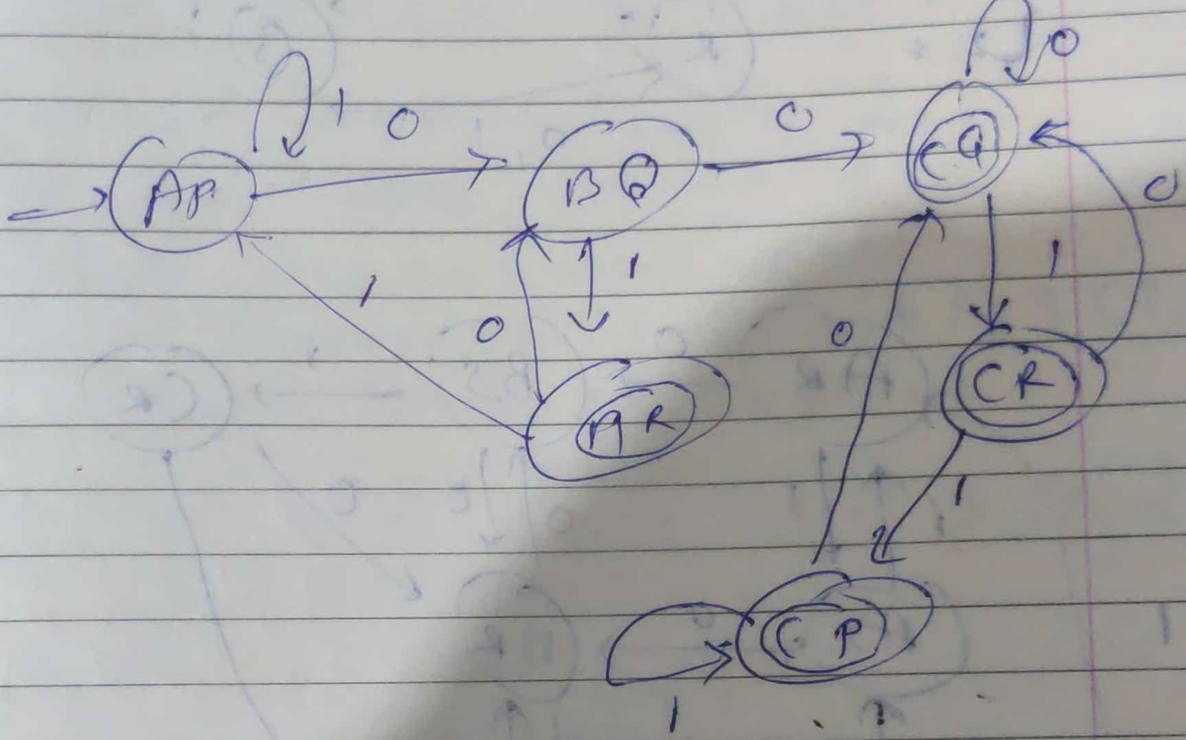
U

end with 01

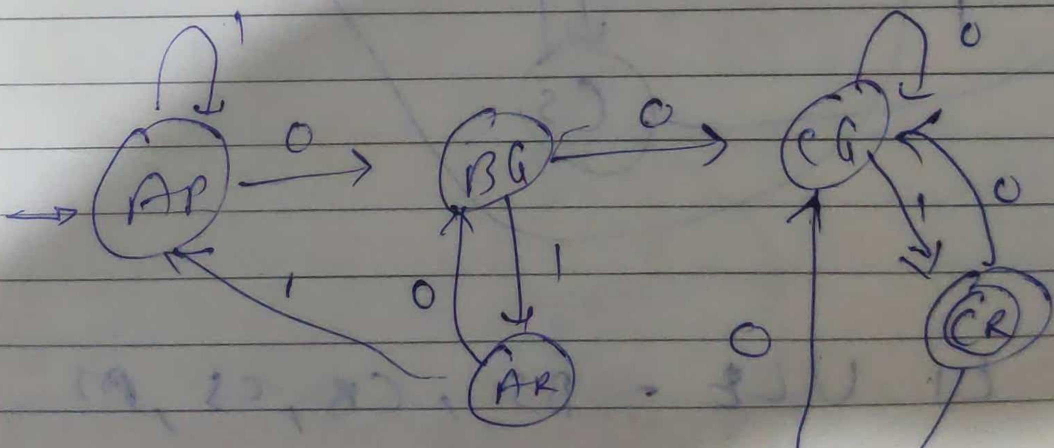
L2



$L1 \cup L2$

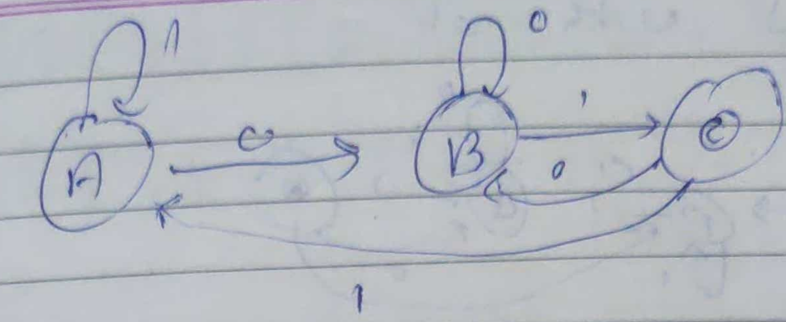


$L1 \cap L2$

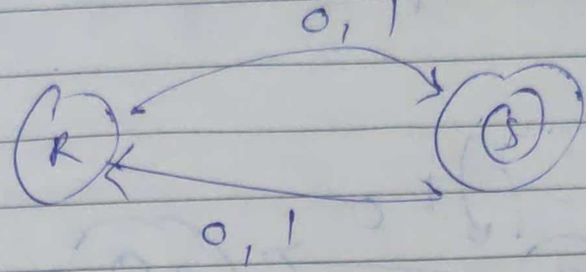


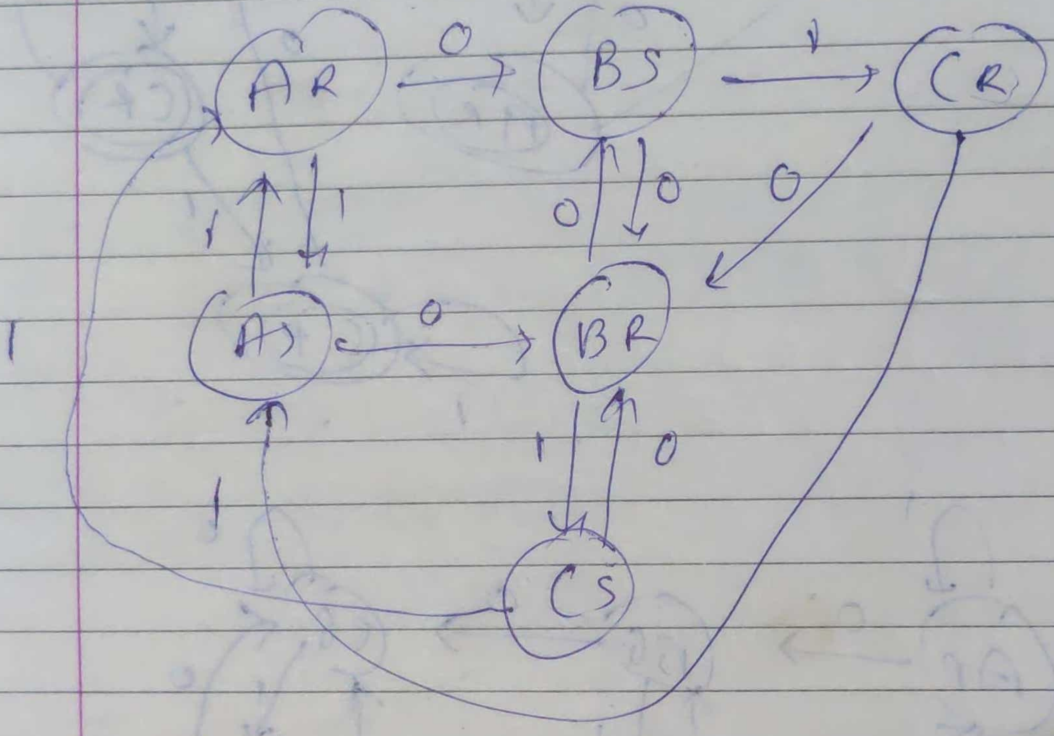
L2-L1 = R will be there but C not so AR become Acepting state.

4 L1 =



L2 =





L1 U L2 = BS , CR , CS , AS

L1 ∩ L2 = CS

L1 - L2 = CR

L2 - L1 = BS , AS

If A is a Regular Language, then A has a Pumping Length 'P' such that any string 'S' where $|S| \geqslant P$ may be divided into 3 parts $S = x\,y\,z$ such that the following conditions must be true:

(1) $x\,y^i z \in A$ for every $i \geqslant 0$

(2) $|y| > 0$

(3) $|xy| \leqslant P$

To prove that a language is not Regular using PUMPING LEMMA, follow the below steps:

(We prove using Contradiction)

-> Assume that A is Regular

-> It has to have a Pumping Length (say P)

-> All strings longer than P can be pumped $|S| \geqslant P$

-> Now find a string 'S' in A such that $|S| \geqslant P$

-> Divide S into $x\,y\,z$

-> Show that $x\,y^i z \notin A$ for some i

-> Then consider all ways that S can be divided into $x\,y\,z$

(1) $x y^i z \in A$ for every $i \geq 0$

(2) $|y| > 0$

(3) $|xy| \leq P$

To prove that a language is not Regular using PUMPING LEMMA, follow the below steps:

(We prove using Contradiction)

-> Assume that A is Regular
-> It has to have a Pumping Length (say P)
-> All strings longer than P can be pumped $|S| \geq P$
-> Now find a string 'S' in A such that $|S| \geq P$
-> Divide S into x y z
-> Show that $x y^i z \notin A$ for some i
-> Then consider all ways that S can be divided into x y z
-> Show that none of these can satisfy all the 3 pumping conditions at the same time
-> S cannot be Pumped == CONTRADICTION

Pumping Lemma (For Regular Languages) - EXAMPLE (Part-1)

Using Pumping Lemma prove that the language $A = \{a^n b^n \mid n \geqslant 0\}$ is Not Regular

Proof:

Assume that A is Regular

Pumping length = p

$S = a^p b^p \Rightarrow S = aaaaaaabbbbbbb$

x y z

p = 7

Case 1: The Y is in the 'a' part

$\underbrace{aa}_{x}\underbrace{aaaa}_{y}\underbrace{abbbbbbb}_{z}$

$xy^iz \Rightarrow xy^2z$ ✗

aa aaaa aaaa a bbbbbbb

$11 \neq 7$

$p = 7$

Case 1: The Y is in the 'a' part

$\underbrace{aa}_{x}\underbrace{aaaa}_{y}\underbrace{a\,bbbbbbb}_{z}$

Case 2: The Y is in the 'b' part

$\underbrace{aaaaaaa\,bb}_{x}\underbrace{bbbb}_{y}\underbrace{b}_{z}$

Case 3: The Y is in the 'a' and 'b' part

$\underbrace{aaaaa}_{x}\underbrace{aabb}_{y}\underbrace{bbbbb}_{z}$

$xy^1z \Rightarrow xy^2z$ ✗

aa aaaa aaaa a bbbbbbb

$11 \neq 7$

$xy^1z \Rightarrow xy^2z$ ✗

aaaaaaa bb bbbb bbbb b

$7 \neq 11$

$xy^1z \Rightarrow xy^2z$ ✗

aaaaa aabbaabb bbbbb

$a^n b^n$

$|xy| \leq p \quad p=7$

Pumping Lemma (For Regular Languages) EXAMPLE (Part-2)

Using Pumping Lemma prove that the language $A = \{yy \mid y \in \{0,1\}^*\}$ is Not Regular

$\underbrace{01}\underbrace{01}$

Proof:

Assume that A is Regular

Then it must have a Pumping Length = P

$S = 0^P 1 0^P 1$

x y z

P = 7

$\underbrace{000}_{x}\underbrace{0000}_{y}\underbrace{100000001}_{z}$

$xy^iz \Rightarrow xy^2z$

00000000000100000001

$\notin A$

$|y| > 0$

$|xy| \leq P = 7$

A is not Regular